

# Optimal Multicast Smoothing of Streaming Video over the Internet<sup>1</sup>

Subhabrata Sen<sup>1</sup>, Don Towsley<sup>2</sup>, Zhi-Li Zhang<sup>3</sup>, and Jayanta K. Dey<sup>4</sup>

<sup>1</sup> Networking Research  
AT&T Labs–Research  
sen@research.att.com

<sup>2</sup> Dept. of Computer Science  
University of Massachusetts  
towsley@cs.umass.edu

<sup>3</sup> Dept. of Computer Science  
University of Minnesota  
zhzhang@cs.umn.edu

<sup>4</sup> Knumi Inc.  
dey@knumi.com

## Abstract

A set of applications such as Internet video broadcasts, corporate telecasts, and distance learning require the simultaneous streaming of video to a large population of viewers across the Internet. The high bandwidth requirements and the multi-timescale burstiness of compressed video make it a challenging problem to provision network resources for streaming multimedia. For such applications to become affordable and ubiquitous, it is necessary to develop scalable techniques to *efficiently* stream video to a large number of disparate clients across a heterogeneous internet. In this paper, we propose to *multicast smoothed video* over an application-level overlay network of proxies, and to *differentially cache* the video at the intermediate nodes (proxies) in the distribution tree, in order to reduce the network bandwidth requirements of video dissemination. We formulate the multicast smoothing problem as an optimization problem, and develop an algorithm for computing the set of transmission schedules for the tree that minimize the peak rate and rate variability, given buffer constraints at different nodes in the tree. We also develop an algorithm to compute the minimum buffer allocation in the entire tree, such that feasible transmission to all the clients is possible, when the tree has heterogeneous rate constraints. We show through trace-driven simulations that substantial benefits are possible from multicast smoothing and differential caching, and that these gains can be realized even with modest proxy caches.

*Keywords:* multimedia communication, multimedia systems, variable-bit-rate video, bandwidth smoothing, differential caching, multimedia proxy, multicast distribution tree, overlay networks

<sup>1</sup>The work at the University of Massachusetts was supported in part under National Science Foundation grants NCR-9523807, NCR-9508274 and CDA-9502639. The works of Subhabrata Sen and Jayanta K. Dey were performed when they were at the University of Massachusetts. Zhi-Li Zhang was supported by a University of Minnesota Graduate School Grant-in-Aid grant, by the NSF CAREER Award Grant NCR-9734428, and NSF ANI 9903228. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding agencies.

## I. INTRODUCTION

The ability to stream continuous media from servers to clients across the Internet enables a wide range of applications. Examples include information and entertainment services (e.g., sportscasts, newscasts), distance education, streaming video broadcasts, corporate telecasts, narrowcasts etc. High quality digital video typically exhibits high transmission bandwidth requirements even when compressed (4 – 6 Mbps for full motion MPEG-2) and significant burstiness at multiple time scales [1, 2], owing to the encoding schemes and the content variation between and within video scenes. When combined with other media such as audio, text and images, multimedia transmissions can become even burstier. Due to the high and bursty bandwidth requirements, coupled with the long-lived characteristics (tens of minutes to a couple of hours) of high quality video, network bandwidth is a major limiting factor in the widespread dissemination of such content over the Internet. As a result, video clips on the Internet today are predominantly low-bandwidth, low quality, and of short duration. In addition, many streaming applications require the transmission of a video to multiple simultaneous users across the network. The challenge of supporting such one-to-many streaming video delivery is further complicated by the heterogeneous nature of clients and the Internet itself. Different portions of an end-end path (Fig. 1) from the server to a client may traverse different ISPs, possess different bandwidth capacities, and experience different loads. Clients themselves are heterogeneous, with varying network access bandwidths and processing and storage capabilities. In this setting, an important question is how to reduce the bandwidth requirements of streaming VBR video for applications like Internet video broadcast and pay-per-view services. *In this paper, we address the problem of streaming prerecorded variable-bit-rate (VBR) video to a large population of heterogeneous clients over the heterogeneous Internet while making efficient use of network resources, and allowing clients to start playback with low delays.* We present a novel technique integrating *workahead smoothing* [3–5] with application-level multicasting and temporal caching to efficiently stream VBR video from a server to multiple heterogeneous clients across a heterogeneous network, using an application-level distribution tree of proxies as shown in Fig. 1. Before presenting the main contributions, we first discuss some key issues that need to be addressed.

### A. Issues and Challenges

A key challenge in delivering video data across the Internet is to reduce high startup delays, and make efficient use of network bandwidth. For Web objects, service providers reduce response time, server load, and network traffic by deploying proxy caches. A proxy cache stores recently accessed resources in the hope of satisfying future client requests without contacting the server. However, video files can be very large, and caching techniques [6–8] for text and image resources that cache entire objects, are not appropriate for the rapidly growing number of continuous media streams in the Internet. Storing the entire contents of several long streams would exhaust the capacity of a conventional proxy cache. Instead, a number of caching strategies have been proposed [9–15] that cache a portion of a video at the proxy. The proxy streams the locally cached portion to the client, while the remainder is streamed from the server.

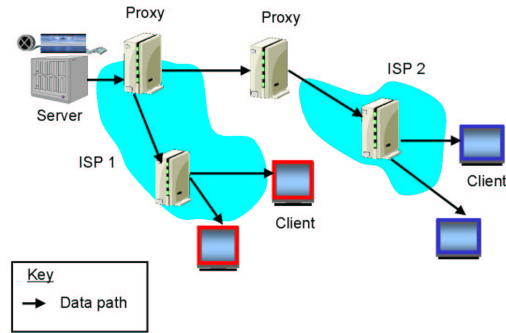


Fig. 1. **Multicast Smoothing in an internet:** A video stream originates at a multimedia server, and travels through the internet, to multiple clients, including workstations and set-top boxes. Multicast smoothing service is performed at smoothing nodes within the network.

Another challenge regards how to efficiently transmit VBR traffic across the network, while achieving high network resource utilization. VBR encoding offers some important advantages over constant-bit-rate (CBR) encoding. For instance, for the same average bandwidth, VBR encoding offers higher quality and a greater opportunity for statistical multiplexing gains than would be possible with a constant-bit-rate (CBR) encoding [2, 16]. However, transmission of VBR video requires effective techniques for transporting bursty traffic across the network. One technique, *unicast workahead smoothing* [3–5, 17] can yield significant reductions in the peak rate and rate variability of video transmissions on the end-to-end network delivery path from a server to a single client (i.e., unicast) within an internet. A characteristic of the smoothed transmission schedule is that the smoothing benefit is a non-decreasing function of buffer size present on the end-to-end video delivery path. The one-to-many streaming scenario we consider, where clients and network paths are heterogeneous, complicates the problem of delivering VBR streaming video.

Finally, the lack of native multicast capability in the network makes it a challenging proposition to support one-to-many high quality video distribution in a scalable, network resource efficient manner. In the early 1990s, a technology called IP multicast [18] was developed to extend the Internet’s original point-point unicast delivery service to offer native network-level point-to-multipoint multicast packet service. However, IP multicast deployment has been slow and even today remains severely limited in scope and reach.

### B. Contributions

In the Internet, in addition to the high and variable bandwidth requirements of video, the multimedia server is faced with the problem of simultaneously streaming a high bandwidth and bursty video to a large number of heterogeneous clients that have different resource capacities (e.g. buffer), over end-to-end paths that have different bandwidth capacities. A naive application of unicast smoothing to a distribution tree would only consider the most constrained client buffer or the most constrained link bandwidth for computing the smoothed transmission schedule to every client in the tree. This approach avoids handling the heterogeneity

in the system and cannot take advantage of the presence of additional resources on other paths in the tree.

We develop a novel distribution scheme that integrates *workahead smoothing* with multicast to efficiently stream video from a single server to several (heterogeneous) clients using an application-level distribution tree topology. Our approach generalizes the unicast smoothing solution to the multicast of smoothed streaming video over a distribution tree.

- We propose *differential caching*, a technique for the temporal caching of video frames at intermediate nodes of the distribution tree, in order to allow more workahead smoothing gains along paths with greater buffer or bandwidth, while enabling clients on paths with smaller buffers or more constrained bandwidths to still receive the entire video at rates suited to such paths.
- We develop a set of algorithms that integrate smoothing with differential caching to compute a set of *optimally* smooth transmission schedules for streaming prerecorded video over a distribution tree, that minimizes the peak rate and variability of the video transmission over each link in the tree in the presence of either buffer or bandwidth constraints. The schemes include feasibility algorithms and algorithms for computing schedules. When buffers at the internal nodes are the constraining resources, we present algorithms to check whether there exists a set of feasible optimal multicast schedules to transmit a particular video. We develop an algorithm that computes the set of optimally smooth transmission schedules when a feasible set of buffers exists. When the link bandwidths in the distribution tree are the constraining resources, we present an algorithm that computes the minimal total buffer allocation for all the nodes in the tree, and the corresponding allocation at each node, such that there exists a set of feasible transmission schedules to distribute a particular video.
- We present MPEG-2 trace-driven evaluations that demonstrate that this integrated multicast smoothing and differential caching technique can result in substantial bandwidth savings, even with modestly sized buffers at the proxies. For the *Blues Brothers* trace for example, with just 512 KB buffer per proxy, optimal multicast smoothing reduces the total transmission bandwidth requirements in the distribution tree by more than a factor of six as compared to multicasting the unsmoothed stream.

### C. Related Work

There has been considerable work in the area of unicast smoothing [3–5]. By transmitting frames early, the sender (or a smoothing node) can send large video frames at a slower rate without disrupting continuous playback at the client. The frames transmitted ahead of time are temporarily stored in buffers present in the server, the client, and any intermediate network nodes. Smoothing can substantially reduce resource requirements under a variety of network service models such as peak-rate based reservation, and bandwidth renegotiation [3]. In the context of video caching, storing an initial prefix of the video [14] at the proxy has numerous applications, e.g., shielding clients from delays and jitter on the server-proxy path, performing online smoothing to reduce the burstiness of VBR video without introducing additional client playback delays, and reducing traffic on the server-proxy path. In this paper, we combine workahead smoothing with

application-level, application-aware multicast and *differential caching* at the proxies.

The notion of multicast presented in this paper is somewhat different from traditional network-level IP multicast, and involves additional functionality at the proxy compared to simple application-level multicast. Our approach utilizes application-level information such as video frame sizes, and system resource availability, and streams video to clients in real-time for continuous playback. More recently other works have proposed application-level multicast with data being delivered over a distribution tree of proxy servers, where a proxy copies a single incoming packet to all its outgoing connections, In the context of Content Distribution Networks (CDNs), Inktomi’s Media Distribution Network [19] uses such an approach. In both IP multicast and simple application-level multicast, internal nodes are only able to forward one copy of every relevant IP packet on each downstream path. These techniques by themselves are not concerned with the real-time nature of the data or with maintaining streaming playback for all the clients. There has been also been work on the use of application level-framing e.g., RTP [20] for video multicast. Our approach complements these different efforts. In our scheme, in addition to packet duplication and forwarding, a node in the distribution tree also performs transmission schedule computation, differential caching, and real time streaming of video according to smoothed transmission schedules. In contrast, IP multicast and simple application-level multicast currently lack support for handling system heterogeneity (either in the clients or the network). Our application-aware approach can be implemented on top of network level multicast primitives, where they exist, and can use unicast communication between nodes in the distribution tree elsewhere. Finally, by integrating smoothing with multicasting, our approach shows far superior performance with respect to reducing the network bandwidth requirements, compared to multicasting the unsmoothed video, as shown in Section VI.

There exists a set of techniques [21–27] that serve clients with heterogeneous resource constraints using either (a) multiple separate encodings on separate multicast groups, with constrained clients receiving a lower quality, lower bandwidth encoding, or (b) hierarchical layered encoding schemes, where each client receives a base layer stream and a variable number of enhancement layer streams. There is some evidence [25, 26] that the first approach is sometimes more useful than the second. Our work is complementary to these techniques.

Numerous papers have focused on the design of scalable video delivery schemes [28–34] that use multicast and broadcast connections to reduce server and network loads. Our smoothing techniques are applicable to all of them.

The remainder of the paper is organized as follows. Section II provides a general problem overview, presents a high level description of our solution approach, and introduces the notion of differential caching. Section III presents the formal model and problem formulations, and Section IV provides a brief overview on smoothing prerecorded video in a unicast setting. Section V develops the solutions to the multicast smoothing problem discussed above. Section VI evaluates our optimal multicast smoothing algorithm and Section VII concludes the paper.

## II. GENERAL PROBLEM DESCRIPTION

An overlay distribution tree consists of a server that injects the video onto the tree, a distribution tree of intervening proxy servers, and end-clients at the leaves of the tree. Fig. 1 illustrates an example architecture. In general, the root of the distribution tree may or may not coincide with the source content server. For example, a proxy or gateway server in a corporate intranet, or a cable headend may receive streaming video from the remote content server, and simultaneously stream out the video to its child nodes on the tree. Each internal node (or proxy) receives video frame data from its parent and transmits them to each of its children. In our scheme, the node also performs other application-specific functions described later.

The distribution tree itself can span multiple network domains. The network connections between consecutive nodes in such an overlay can be either (i) fixed bandwidth connections such as leased lines or VPN (Virtual Private Network) connections, or (ii) regular Internet connections that are shared with all users <sup>1</sup>.

Different parts of the overlay network may have different resource constraints. Available link bandwidths can vary and the overlay nodes may differ in their available buffer sizes. Clients may have different connectivity to the network, e.g., a client could be connected via a slow modem or a high speed LAN. A client can be a workstation, PC, hand-held multimedia device or a set-top box connected to a television set, thus possessing varying buffering and computational resources.

The key question we ask in this paper is how to smooth transmissions and efficiently deliver video data over such a heterogeneous network, where either the links are bandwidth constrained or the nodes and clients are buffer constrained. Given the available buffer at the proxies and clients, we want to transmit the video as smoothly as possible along each link, without underflowing or overflowing the available buffers. When the links have bandwidth constraints, it may be necessary to build up sufficient data at various proxies and clients before clients can start playback, to guarantee starvation-free playback at each client. Given the bandwidth availability on the different links, our goal is to (i) determine the minimum buffer allocation required at each node in the tree and the corresponding minimum playback startup delay for all clients such that it is possible to guarantee starvation-free playback at the clients, and (ii) transmit the video according to the smoothest possible schedule along each link.

The following are some desirable properties of a one-to-many streaming video distribution scheme :

- *Low startup delays*: Clients should be able to start playback with low delays.
- *Bandwidth Scalability*: The distribution scheme should make efficient use of the end-end network bandwidths. This benefits the CDN, the clients's local ISP, and finally the end-clients by reducing transmission bandwidth usage costs, enabling the overlay network to handle a larger client population with the same total provisioned bandwidth.

<sup>1</sup>The video distribution tree could be realized in a number of other ways. In an active network [35] setting, the internal nodes in the tree can be switches or routers in the network. In a cable network setting, the head-end and mini-fiber nodes in the network would be natural candidates for hosting the distribution tree node functionality.

As part of our smoothing solution, we introduce the concept of differential caching at a proxy. We propose buffering the video stream partially at the root and intermediate nodes of the distribution tree in order to smooth the streaming clip. Buffer availability at a node allows temporal caching of portions of the video streams. We refer to this technique as *differential caching*, which can be of tremendous benefit for streaming video in a heterogeneous internet. Caching at the root node allows it to smooth an incoming live (or stored) video stream, and transmit the resultant schedule to a downstream node. The buffers at the internal nodes are used for several functions. First, these buffers allow a node to accommodate the differences between its incoming and outgoing transmission schedules when the outgoing and incoming link capacities differ. Second, as described in later sections, by increasing the *effective* or *virtual* smoothing buffer size for the upstream node, these buffers provide more aggressive smoothing opportunities along the upstream link. Finally, when the children of a node have different effective buffer capacities, the smoothed transmission schedules differ along the different outgoing links. For a given incoming transmission schedule, the node temporally caches video frames until they are transmitted along each outgoing link. Thus differential caching allows the transmission schedule to each child to be decoupled to varying degrees depending on the size of the parent's buffer cache. This can be extremely beneficial from a network resource requirements point of view. For example, the parent node can smooth more aggressively to a child which has a larger (effective) buffer relative to a child with a smaller buffer. Differential caching allows the more constrained child to be served at a pace more suitable to it, without requiring the frames to be retransmitted from higher up in the tree.

Given a heterogeneous distribution tree where the intermediate nodes can perform differential caching, the important questions we need to address include (a) How to allocate resources to the intermediate nodes in the distribution tree so that streaming transmission to all the clients is possible? (b) Given a particular resource allocation to the distribution tree, what transmissions schedules should be used for the multicast, so that the bandwidth requirements are minimized? The rest of the paper is devoted to these questions. We begin with the formal problem definition in Section III.

### III. MULTICAST DISTRIBUTION OF SMOOTHED VIDEO

In this section, we first present a formal model for the video multicast distribution tree, and then outline the buffer and rate constrained multicast smoothing problems.

#### A. Video Multicast Distribution Tree Model

Consider a directed tree  $T = (V, E)$  (Fig. 2) where  $V = \{0, 1, \dots, n\}$  is the set of nodes (or proxies) and  $E$  is the set of directed edges within the tree. Consider a node  $i \in V$ , let  $p(i)$  denote its parent and  $s(i)$  the set of children attached to  $i$ , i.e.,  $(p(i), i) \in E$  and  $s(i) = \{j \in V : (i, j) \in E\}$ . We take node 0 to be the root server in the tree. Let  $V_l \subset V$  be the set of leaves;  $V_l = \{i \in V : s(i) = \emptyset\}$ . Note that node 0 could itself be the source server of the video, or the video may be streamed into the root from a remote source.

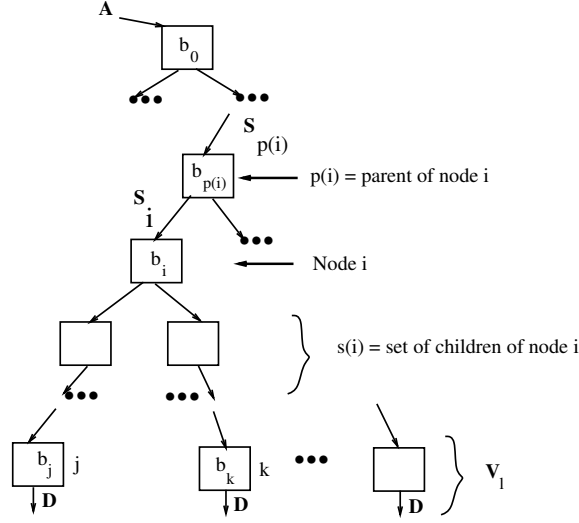


Fig. 2. Multicast Smoothing Model

Also, a leaf can be an end-client, or an egress node in the distribution tree. We will use the two pairs of terms *root* and *source*, and *leaf* and *client* interchangeably.

Associated with node  $i \in V$  is a buffer of capacity  $b_i$ ,  $0 \leq b_i < \infty$ . Without loss of generality, we assume a discrete time model where one time unit corresponds to the time between successive frames (33 msec for 30 frames/second full motion video). Consider an  $N$ -frame video stream (where the size of frame  $i$  is  $f_i$  bits,  $1 \leq i \leq N$ ) which arrives at node 0 destined for all nodes  $i \in V_l$ . Associated with this video stream is an *arrival vector*  $\mathbf{A} = (A_0, \dots, A_N)$  in which the  $k$ -th component corresponds to the cumulative amount of data from the stream which has arrived at node 0 by time  $k = 0, 1, \dots, N$ . It is assumed that  $A_k \geq A_{k-1}$ ,  $1 \leq k \leq N$ . For each node  $i \in V_l$ ,  $\mathbf{D} = (D_0, \dots, D_N)$  denotes the *client playback vector* where  $D_k$  represents the cumulative amount of data from this stream that must be removed at leaf  $i$  by  $k = 0, 1, 2, \dots, N$  time units since the start of playback. It is assumed that  $D_0 = 0$  and  $D_k \geq D_{k-1}$ ,  $1 \leq k \leq N$ . For example,  $D_k = \sum_{j=1}^k f_j$ , if the output from the leaf is the original unsmoothed video. For simplicity of exposition, we assume that all clients have the same playback vector. The more general setting where clients have different playback vectors is dealt with in an extended version of this work [36]. We will refer to the case where the root server has an “infinite” buffer, in the sense that  $b_0 \geq D_N$ , as the *infinite source* model, and the case where  $b_0 < D_N$  as the *finite source* model.

Associated with each node  $i \in V \setminus \{0\}$ <sup>2</sup> is a *schedule*  $\mathbf{S}_i = (S_{i,0}, \dots, S_{i,N})$  in which the  $k$ -th component denotes the cumulative amount of data transmitted by node  $p(i)$  to node  $i$  by time  $k = 0, 1, \dots, N$ ,  $1 \leq i \leq n$ . Note that  $\mathbf{S}_0 \equiv \mathbf{A}$ , and  $S_{i,k} \geq S_{i,k-1}$ ,  $1 \leq k \leq N$ . A schedule set  $\{\mathbf{S}_i\}_{i \in V}$  is said to be *feasible* if the simultaneous use of the component schedules for video distribution over the tree does not violate any system constraints, and results in lossless, starvation-free playback at each leaf. We shall provide a formal definition of feasibility later in the Section.

<sup>2</sup>The expression  $P \setminus Q$  refers to all elements of set  $P$  except those belonging to set  $Q$ .

In general, the time when the root server starts transmitting a video stream (or the time when the video stream begins arriving at the root) may differ from the time when a client starts playing back the video stream. The difference between these two start points is referred to as *startup delay*. For a given startup delay  $w \geq 0$ , if we take the time when the server starts transmitting a video stream as the reference point (i.e., time 0), then the playback vector  $\mathbf{D}$  at the client  $i$  will be shifted  $w$  time units to the right. This shifted playback vector is represented by  $\mathbf{D}(w) = (D_0(w), D_1(w), \dots, D_{N+w}(w))$ , where  $D_i(w) = 0$  for  $0 \leq i \leq w$ , and  $D_i(w) = D_{i-w}$  for  $w + 1 \leq i \leq N + w$ . In this case, the root has  $N + w + 1$  time units to transmit the video stream, and the schedule is  $\mathbf{S}_j = (S_{j,0}, S_{j,1}, \dots, S_{j,N+w}), j \in s(0)$ . For ease of notation, we will also extend the arrival vector  $\mathbf{A}$  (which starts at time 0) with  $w$  more elements, namely,  $\mathbf{A} = (A_0, \dots, A_{N+w})$ , where  $A_i = A_N$  for  $N + 1 \leq i \leq N + w$ . If we take time 0 to be the instant when the client begins playback, then we shift the arrival vector  $\mathbf{A}$   $w$  time units to the left. The corresponding arrival vector is denoted by  $\mathbf{A}(-w) = (A_{-w}(w), A_{-w+1}(w), \dots, A_N(w))$  where for  $-w \leq i \leq N - w$ ,  $A_i(w) = A_{i+w}$ , and  $A_i(w) = A_N$  for  $N - w + 1 \leq i \leq N$ . Similarly, the root starts transmission at time  $-w$  according to schedule  $\mathbf{S}_j = (S_{j,-w}, \dots, S_{j,0}, \dots, S_{j,N})$ . In the rest of the paper, we will define the time that the root server starts transmission as 0 unless otherwise stated. Depending on the context,  $\mathbf{A}$  (or  $\mathbf{D}$ ) will denote either the generic arrival (or playback) vector or the appropriately shifted version.

### B. Buffer Constrained Optimal Multicast Smoothing

Given a set of buffer allocations  $\{b_i\}_{i \in V}$ , and consumption vector  $\mathbf{D}$  at the leaves of the distribution tree, let  $\mathcal{S}(T, \{b_i\}, \mathbf{A}, \mathbf{D})$  denote the set of all feasible schedule sets for this system. In this context, the set  $\{\mathbf{S}_i\}_{i \in V}$  is feasible if

$$\max\{\mathbf{S}_{p(i)} - \text{vec}(b_{p(i)}), \max_{j \in s(i)} \mathbf{S}_j\} \leq \mathbf{S}_i \leq \min\{\min_{j \in s(i)} \mathbf{S}_j + \text{vec}(b_i), \mathbf{S}_{p(i)}\}, \quad i \in V \setminus V_l, \quad (1)$$

and

$$\max\{\mathbf{S}_{p(i)} - \text{vec}(b_{p(i)}), \mathbf{D}\} \leq \mathbf{S}_i \leq \min\{\mathbf{D} + \text{vec}(b_i), \mathbf{S}_{p(i)}\}, \quad i \in V_l, \quad (2)$$

where  $\mathbf{S}_0 \equiv \mathbf{A}$  and  $\text{vec}(a)$  denotes a vector whose components are all equal to  $a$ .

Intuitively, at any time  $k$ , the cumulative amount of data  $S_{i,k}$  arriving at node  $i$  should be sufficient to satisfy the needs of all its children, and cannot exceed the cumulative amount of data being received at its parent  $p(i)$ . Also,  $\mathbf{S}_i$  should transmit data fast enough to prevent buffer overflow at  $p(i)$ , but not fill up the buffer at  $i$  so quickly that the transmission ( $\mathbf{S}_j, j \in s(i)$ ) to some child is unable to transfer data from the buffer in time, before it gets overwritten.

The following inequalities follow almost immediately,

$$\mathbf{S}_i \leq \mathbf{S}_{p(i)} \leq \mathbf{S}_i + \text{vec}(b_i), \quad i \in V \setminus \{0\}. \quad (3)$$

Two important questions in this setting are

1. *Buffer feasibility* : Is the buffer allocation feasible, i.e., is the set of feasible schedules  $\mathcal{S}(T, \{b_i\}, \mathbf{A}, \mathbf{D})$  nonempty? Note that the feasibility question arises as all the buffer allocations are assumed to be finite. Given a particular arrival vector  $\mathbf{A}$  and playback vector  $\mathbf{D}$  at the leaves, it is possible that no feasible transmission schedule is possible at one or more edges in the tree, due to buffer overflow at the source or sink of that edge.
2. *Optimal smoothed schedules* : For a feasible buffer allocation, what is the optimal set of smoothed transmission schedules  $\{\mathbf{S}_i\}_{i \in V}$ ?

### C. Rate Constrained Optimal Multicast Smoothing

Here we assume that the distribution tree is bandwidth constrained rather than buffer constrained. We only consider the *infinite source* model, i.e.,  $b_0 \geq D_N$ .

Link  $(p(i), i)$  has a maximum transmission rate of  $r_i$ ,  $i \in V \setminus \{0\}$ .  $\mathbf{S}_i = (S_{i,0}, S_{i,1}, \dots)$  denotes a schedule used by node  $p(i)$  to transmit data on the link  $(p(i), i)$ , where  $S_{i,k}$  is the cumulative amount of data transmitted by time  $k$ . Define the peak rate of  $\mathbf{S}_i$  to be  $peak(\mathbf{S}_i) = \max_k \{S_{i,k} - S_{i,k-1}\}$ . We say  $\{\mathbf{S}_i\}_{i \in V}$  is a *feasible* schedule if the following conditions are satisfied:

$$peak(\mathbf{S}_i) \leq r_i, \quad i \in V \setminus \{0\}, \quad (4)$$

$$\max_{j \in s(i)} \mathbf{S}_j \leq \mathbf{S}_i \leq \mathbf{S}_{p(i)}, \quad i \in V - V_l,$$

$$\text{or } \mathbf{D} \leq \mathbf{S}_i \leq \mathbf{S}_{p(i)}, \quad i \in V_l. \quad (5)$$

We define  $\mathcal{S}(T, \{r_i\}, \mathbf{A}, \mathbf{D})$  to be the set of feasible transmission schedule sets for this system. We are interested in addressing the following questions.

1. *Minimum startup delay* : What is the minimum playback startup delay  $w^*$  for all the clients for which a feasible transmission schedule exists for the system, i.e.,  $\mathcal{S}(T, \{r_i\}, \mathbf{A}, \mathbf{D}) \neq \emptyset$ ? Note that the client playback startup delay may be greater than zero for the rate constrained scenario. Due to the transmission rate constraints, some minimum startup delay may be required to build up sufficient data in the proxies and at the clients to guarantee starvation-free playback at each client. Here we assume that all clients start playback at the same time.
2. *Optimal buffer allocation* : What is the minimum buffer allocation  $b_i$  at each node  $i \in V \setminus \{0\}$  of the distribution tree, and what is the minimum total buffer allocation  $b = \sum_{i \in V \setminus \{0\}} b_i$  among all feasible schedules for the system? As explained before, the buffer is used for differential caching. Observe that for a given set of feasible schedules  $\{\mathbf{S}_i\}_{i \in V}$ , the buffer allocation  $\{b_i\}_{i \in V \setminus \{0\}}$  for the distribution tree is said to be *feasible* if the following constraints are satisfied:  $b_i$  must be sufficiently large to ensure lossless video transmission, namely, it must be able to accommodate both (a) the maximum difference between the amounts of data transmitted from node  $i$  to any pair of child nodes  $k$  and  $l$  according to feasible schedules  $\mathbf{S}_k$  and  $\mathbf{S}_l$  as well as (b) the maximum difference between the amount of data arriving

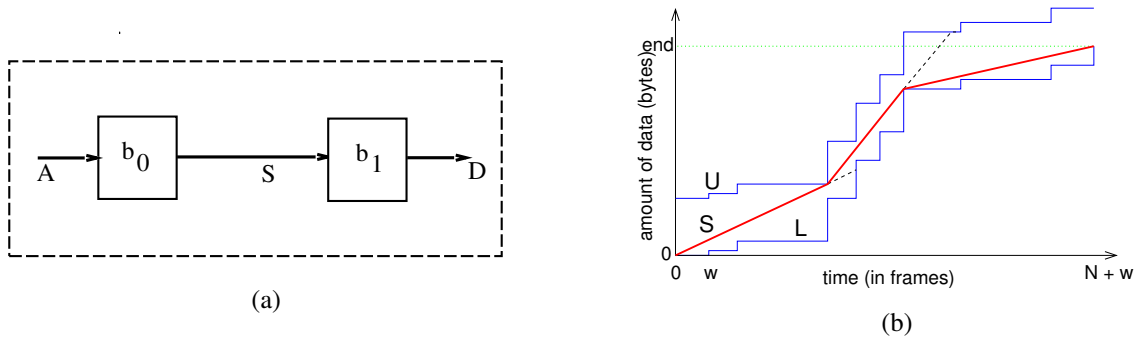


Fig. 3. (a) depicts the Single Link Smoothing Model, and (b) shows an example of a transmission schedule  $\mathbf{S}$  that stays between the upper and lower constraint vectors  $\mathbf{U}$  and  $\mathbf{L}$ .

at node  $i$  according to a schedule  $\mathbf{S}_i$  and that transmitted from node  $i$  to a child node  $k$  according to a schedule  $\mathbf{S}_k$ . Formally,

$$b_i \geq \max_{l, k \in s(i)} \{\max\{\mathbf{S}_l - \mathbf{S}_k\}\}, \quad (6)$$

$$b_i \geq \max_{k \in s(i)} \{\max\{\mathbf{S}_i - \mathbf{S}_k\}\}. \quad (7)$$

#### IV. OVERVIEW OF SINGLE LINK SMOOTHING

This section describes the single link smoothing model, and overviews some important concepts and results which are crucial in deriving solutions for the multicast scenario.

Consider (Fig. 3(a)) a video transmitted across the network via a *smoothing server* node [5] (which has a  $b_0$  bit buffer) to a *smoothing client* node (which has a  $b_1$  bit playback buffer). The formal model for this single-link case can be derived from the distribution tree model in Section III-A by setting  $n = 1$ . Consider next the buffer and rate constrained versions of this problem.

##### A. Buffer Constrained Single Link Optimal Smoothing

Here the server and client buffers are the limiting resources, and the smoothing problem involves computing transmission schedules which can stream the video from the server to the client in such a way as to reduce the variability of the transmitted stream, thereby making efficient use of network bandwidth.

To ensure lossless, continuous playback at the client, the server must transmit sufficient data to avoid *buffer underflow* at the client without *overflowing* the server buffer. This imposes a lower constraint,  $L_t = \max\{D_t(w), A_t - b_0\}$ , on the cumulative amount of data that the server can transmit by any time  $t$ ,  $0 \leq t \leq N + w$ . In order to prevent *overflow* of the client playback buffer, the cumulative amount of data received by the client by time  $t$  cannot exceed  $D_{t-1}(w) + b_1$ . This leads to the following upper constraint  $U_t = \min\{D_{t-1}(w) + b_1, A_t\}$ ,  $0 \leq t \leq N + w$ . Given these lower (or *buffer underflow*) and upper (or *buffer overflow*) constraint vectors  $\mathbf{L} = (L_0, \dots, L_{N+w})$  and  $\mathbf{U} = (U_0, \dots, U_{N+w})$ , a transmission schedule  $\mathbf{S} = (S_0, S_1, \dots, S_{N+w})$  is said to be *feasible* with respect to  $\mathbf{L}$  and  $\mathbf{U}$  if  $S_0 = L_0$ ,  $S_{N+w} = L_{N+w}$ , and  $\mathbf{S}$  neither underflows nor overflows the server or client buffer, i.e.,  $\mathbf{L} \leq \mathbf{S} \leq \mathbf{U}$  (see Fig. 3(b)).

In general, for a given pair of constraint vectors  $(\mathbf{L}, \mathbf{U})$  such that  $\mathbf{L} \leq \mathbf{U}$ , multiple feasible transmission schedules  $\mathbf{S}$  may exist. Let  $\mathcal{S}(\{b_i\}, \mathbf{A}, \mathbf{D})$  denote the set of all feasible schedules for the single link system. Among all feasible schedules, we would like to find a ‘‘smoothest’’ schedule that minimizes network utilization according to some performance metrics [4]. In [3], a measure of smoothness based on the theory of *majorization* is proposed, and the resulting *optimally* smoothed schedule minimizes a wide range of bandwidth metrics such as the peak rate, the variability of the transmission rates and the empirical effective bandwidth. Henceforth, we shall refer to this ‘‘optimal schedule’’ (linear time construction) as the *majorization schedule*.

For any two  $K$ -dimensional real vectors  $X = (X_1, \dots, X_K)$  and  $Y = (Y_1, \dots, Y_K)$ ,  $Y$  is said to *majorize*  $X$  (denoted by  $X \prec Y$ ) [37] if  $\sum_{i=1}^K X_i = \sum_{i=1}^K Y_i$ , and for  $k = 1, \dots, K - 1$ ,  $\sum_{i=1}^k X_{[i]} \leq \sum_{i=1}^k Y_{[i]}$  where  $X_{[i]}$  (resp.,  $Y_{[i]}$ ) is the  $i$ -th largest component of  $X$  (resp.,  $Y$ ). A notion closely associated with majorization is *Schur-convex* function. A function  $\phi : \mathbb{R}^K \rightarrow \mathbb{R}$  is said to be a *Schur-convex* function iff  $X \prec Y \Rightarrow \phi(X) \leq \phi(Y)$ ,  $\forall X, Y \in \mathbb{R}^K$ . Examples of Schur-convex functions include  $\phi(X) = \max_i X_i$ , and  $\phi(X) = \sum_{i=1}^K f(X_i)$  where  $f$  is any convex real function.

In the context of video smoothing, for a transmission schedule  $\mathbf{S} = (S_0, S_1, \dots, S_{N+w})$ , define  $\mathbf{R}(\mathbf{S}) = (S_1 - S_0, \dots, S_{N+w} - S_{N+w-1})$ . A schedule  $\mathbf{S}_1$  is majorized by (or intuitively, ‘‘smoother’’ than) another schedule  $\mathbf{S}_2$  (denoted by  $\mathbf{S}_1 \prec \mathbf{S}_2$ ) if  $\mathbf{R}(\mathbf{S}_1) \prec \mathbf{R}(\mathbf{S}_2)$ . The *majorization schedule*,  $\mathbf{S}^*(\mathbf{L}, \mathbf{U})$ , or in short  $\mathbf{S}^*$ , is the schedule such that  $\mathbf{R}(\mathbf{S}^*) \prec \mathbf{R}(\mathbf{S})$ ,  $\forall \mathbf{S} \in \mathcal{S}(b, \mathbf{A}, \mathbf{D})$ . In particular, the peak rate of  $\mathbf{S}^*$ ,  $peak(\mathbf{S}^*) = \max_k \{S_k - S_{k-1}\}$ , is minimal among all feasible schedules  $\mathbf{S}$ . In [3], it is shown that  $\mathbf{S}^*$  exists and is unique (provided that  $\mathbf{L} \leq \mathbf{U}$ ), and it can be constructed in linear time  $O(N)$ , where  $N$  is the number of frames in the video.

Given a majorization schedule  $\mathbf{S}^*$ , we say  $k$ ,  $1 \leq k \leq N + w - 1$ , is a *change point* of the schedule, if the transmission rate changes at time  $k$ . Moreover,  $k$  is said to be a *convex (concave) change point* if the transmission rate increases (decreases) at  $k$ . A key feature of the majorization schedule is that the convex (concave) change points occur precisely at the times  $k$  such that  $S_k^* = U_k$  ( $S_k^* = L_k$ ). This property leads (proof in [38]) to the following lemma which is critical to constructing optimal smoothing schedules for the distribution tree scenario.

*Lemma 1:* Let  $\mathbf{L}_1 \leq \mathbf{U}_1$  and  $\mathbf{L}_2 \leq \mathbf{U}_2$  be such that  $\mathbf{L}_1 \leq \mathbf{L}_2$  and  $\mathbf{U}_1 \leq \mathbf{U}_2$ . Then  $\mathbf{S}_1^* \leq \mathbf{S}_2^*$ . Moreover, if  $\mathbf{L}_1 = \mathbf{L}_2$ , then any concave (convex) change point of  $\mathbf{S}_2$  is also a concave (convex) change point of  $\mathbf{S}_1$ .

In particular, we have

*Lemma 2:* For any  $b_2 \geq b_1 > 0$  and  $\mathbf{L}_1 = \mathbf{L}_2$ , define  $\mathbf{U}_1 = \mathbf{L}_1 + vec(b_1)$  and  $\mathbf{U}_2 = \mathbf{L}_2 + vec(b_2)$ . Then the set of the change points of  $\mathbf{S}_1^*$  is a superset of the change points of  $\mathbf{S}_2^*$ .

## B. Rate Constrained Single Link Optimal Smoothing

We next consider the dual problem of the *rate constrained* single link optimal smoothing, where the bandwidth of the link connecting the server and the client is constrained by a given rate  $r$ . Due to the rate

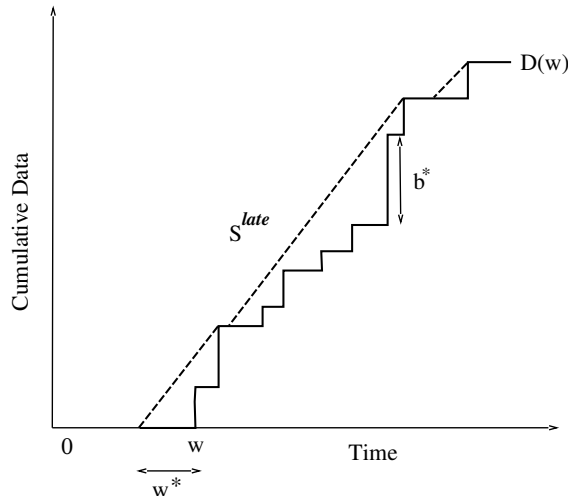


Fig. 4. This figure depicts the *lazy* transmission schedule  $\mathbf{S}^{late}$ .  $b^*$  and  $w^*$  are the respective minimum buffer and minimum startup delays necessary to satisfy rate constraint  $r$ .

constraint, the buffer at the client must be *sufficiently large* to ensure continuous video playback at the client. Furthermore, it may be necessary for the server to start video transmission *sufficiently early*. In this context, a server transmission schedule is *feasible* if the transmission rate of the schedule never exceeds the rate constraints at any time (as well as the arrival vector) and the amount of data needed for the client playback is always satisfied at any time.

We assume that the client playback starts at time 0, and as before, the cumulative client playback vector is denoted by  $\mathbf{D}(0) = (D_0(0), D_1(0), \dots, D_N(0))$ . Then, given a startup delay  $w$ , we assume that the arrival vector starts at time  $-w$  instead of time 0, with the shifted arrival vector  $\mathbf{A}(-w)$  (see notation in Section III-A). Consider the *infinite source* model. It is possible to construct, in linear time, a *feasible* transmission schedule  $\mathbf{S}^{late}$  [39–41] which transmits data as late as possible, while obeying rate constraint  $r$ . We refer to this as the *lazy schedule* ( Fig.4).

Define  $b^*(r)$  to be the minimum buffer required at the client for the transmission schedule  $\mathbf{S}^{late}$  without incurring loss of data, and  $w^*(r, \mathbf{A})$ , the minimum start-up delay with respect to which  $\mathbf{S}^{late}$  conforms to the arrival vector, i.e.,  $S_k^{late} \leq A_k(-w^*)$ , for  $k \geq w^*$ . Then,

$$b^*(r) = \max_{k \geq 0} \{S_k^{late} - D_k(0)\}, \quad (8)$$

$$w^*(r, \mathbf{A}) = \min\{w \geq 0 : A_k(-w) - S_k^{late} \geq 0, -w \leq k \leq N\}. \quad (9)$$

It can be shown that  $b^*$ <sup>3</sup> is the minimal buffer requirement and  $w^*$  is the minimal start-up delay among all feasible schedules with respect to the given rate constraint  $r$  and arrival vector  $\mathbf{A}$ . Also  $b^*(r)$  and  $w^*(r, \mathbf{A})$  are nonincreasing functions of the rate constraint  $r$ .

<sup>3</sup>We will write  $b^*$  for  $b^*(r)$  and  $w^*$  for  $w^*(r, \mathbf{A})$  whenever there is no danger of confusion.

The next theorem (proof in Appendix) relates the rate-constrained optimal smoothing problem to its dual buffer-constrained optimal smoothing problem. For ease of exposition, choose time 0 as the time  $A_0$  arrives at the server and the server starts video transmission. Corresponding to any playback startup delay  $w \geq w^*$ , the client playback vector is  $\mathbf{D}(w)$ . The new lazy schedule  $\mathbf{S}^{late}$  is then the original  $\mathbf{S}^{late}$  shifted  $w$  time units to the right. By the definition of  $w^*$ , we have  $w^* = \min\{w > 0 : A_k - S_k^{late} \geq 0, 0 \leq k \leq N + w\}$ .

*Theorem 1:* Let  $\mathcal{S}(r, \mathbf{A}, \mathbf{D})$  be the set of all feasible schedules with respect to the rate constraint  $r$  and the arrival vector  $\mathbf{A}$ . Let  $b^*(r)$  and  $w^*(r, \mathbf{A})$  be the minimum client buffer requirement and startup delay of  $\mathcal{S}(r, \mathbf{A}, \mathbf{D})$ . For any  $w \geq w^*(r, \mathbf{A})$ , define  $\mathbf{L} = \mathbf{D}(w)$  and  $\mathbf{U} = \min\{\mathbf{A}, \mathbf{D}(w) + \text{vec}(b^*(r))\}$ . Then the majorization schedule  $\mathbf{S}^*$  with respect to the buffer-constraints  $\mathbf{L}, \mathbf{U}$  is also a feasible schedule with respect to the rate-constraint  $r$ , and  $\text{peak}(\mathbf{S}^*) = \text{peak}(\mathbf{S}^{late})$ . In particular, if  $\text{peak}(\mathbf{S}^{late}) = r$ , then  $\text{peak}(\mathbf{S}^*) = r$ .

As a consequence of Theorem 1, we see that for  $w = w^*$ , the majorization schedule  $\mathbf{S}^*$  is majorized (thus “smoothest” under the measure of majorization) by any feasible schedule in  $\mathcal{S}(r, \mathbf{A}, \mathbf{D})$  which has the *same* client buffer requirement  $b^*(r)$  and startup delay  $w^*(r, \mathbf{A})$ .

In the following, we establish an important property (proof in Appendix) for the majorization schedule in the context of the rate constrained smoothing problem. This will be useful for computing the optimal buffer allocation in the rate-constrained multicast scenario.

*Lemma 3:* Given two rate constraints  $r_i, i = 1, 2$ , where  $r_1 \geq r_2$ , let  $b_i^* = b^*(r_i)$  and  $w_i^* = w^*(r_i, \mathbf{A})$  be the corresponding minimum client buffer requirement and startup delay with respect to the rate constraint  $r_i$  and an arrival vector  $\mathbf{A}$ . For any  $w \geq \max\{w_1^*, w_2^*\}$ , define  $\mathbf{L}_i = \mathbf{D}(w)$  and  $\mathbf{U}_i = \min\{\mathbf{A}, \mathbf{D}(w) + \text{vec}(b_i^*)\}, i = 1, 2$ . Let  $\mathbf{S}_i^*$  be the majorization schedule with respect to  $(\mathbf{L}_i, \mathbf{U}_i), i = 1, 2$ . Then

$$\mathbf{S}_1^* \leq \mathbf{S}_2^* \text{ and } \max\{\mathbf{S}_2^* - \mathbf{S}_1^*\} = b_2^* - b_1^*. \quad (10)$$

Moreover, for any feasible schedule  $\mathbf{S}_i$  such that  $\mathbf{L}_i \leq \mathbf{S}_i \leq \mathbf{U}_i$  and  $\text{peak}(\mathbf{S}_i) \geq \text{peak}(\mathbf{S}_i^*)$ , we have

$$\max\{\mathbf{S}_2 - \mathbf{S}_1\} \geq b_2^* - b_1^*. \quad (11)$$

## V. OPTIMAL MULTICAST SMOOTHING

In this section we develop solutions to the multicast problems outlined in Sections III-B and III-C. We utilize the results for the single link smoothing problem to build our solutions for the multicast problem. A key step in our approach involves computing upper and lower constraint curves and exploiting the properties of majorization and lazy transmission schedules. We will present relevant theoretical results whose proofs are provided in the Appendix.

### A. Buffer Constrained Optimal Multicast Smoothing

Analogous to the single link case, we first compute upper and lower constraint curves at the individual nodes in the distribution tree. Unlike the single link case, the constraints at a node can be affected by both the

constraints at its directly connected nodes *as well as* at remote nodes. Then, we shall present some important results based on which we develop algorithmic solutions to the two questions posed in Section III-B.

*Upper Constraint:*

For  $i \in V$ , we define a vector  $\mathbf{U}_i^b$  recursively as follows.

$$\mathbf{U}_i^b = \begin{cases} \mathbf{D} + \text{vec}(b_i) & \text{for } i \in V_l \\ \min_{j \in s(i)} \mathbf{U}_j^b + \text{vec}(b_i) & \text{for } i \in V \setminus V_l \end{cases} \quad (12)$$

$\mathbf{U}_i^b$  can be thought of as the *buffer overflow* (or *upper*) *constraint vector* for node  $i$  when it is fed by the source of the prerecorded video, i.e.,  $A_k = D_N, k = 0, \dots, N$ . We will see shortly that it is possible for a buffer constraint somewhere upstream in the tree to impose a more stringent constraint than  $\mathbf{U}_i^b$  on the transmission schedules for the subtree rooted at node  $i$ .

For  $i \in V$ , let  $P(i)$  denote the set of nodes on the path from the root, 0, to node  $i$  and define

$$\mathbf{U}_i^c = \min_{j \in P(i)} \mathbf{U}_j^b. \quad (13)$$

We will refer to  $\mathbf{U}_i^c$  as the *effective buffer overflow constraint vector* of the subtree rooted at  $i$ . Observe that  $\mathbf{U}_i^c \leq \mathbf{U}_i^b$ . The effective overflow vectors exhibit the following properties.

$$\mathbf{U}_i^c \leq \mathbf{U}_{p(i)}^c \leq \mathbf{U}_i^c + \text{vec}(b_{p(i)}). \quad (14)$$

*Lower Constraint:*

Also associated with node  $i$  is an *effective buffer underflow constraint vector*,  $\mathbf{L}_i^c$  defined by the following recurrence relation :

$$\mathbf{L}_i^c = \begin{cases} \mathbf{D}, & i = 0, \\ \max(\mathbf{D}, \mathbf{L}_{p(i)}^c - \text{vec}(b_{p(i)})), & i \neq 0. \end{cases} \quad (15)$$

We now consider a single link system with an arrival vector  $\mathbf{A}$  in which the source has a buffer capacity of size  $G_{p(i)} \equiv \sum_{j \in P(p(i))} b_j$  and the receiver has buffer overflow and underflow constraints  $\mathbf{U}_i^c$  and  $\mathbf{L}_i^c$ . Let  $\mathbf{S}_i^*$  denote the majorization schedule for this system. The following lemma (proof in Appendix) states that the schedules  $\{\mathbf{S}_i^*\}_{i=1, \dots, n}$  are feasible transmission schedules for the buffer constrained multicast scenario. This result hinges on a key property of majorization schedules (Lemma 1).

*Lemma 4:* The schedule  $\mathbf{S}_i^*$  satisfies the following constraints.

$$\max\{\mathbf{S}_{p(i)}^* - \text{vec}(b_{p(i)}), \max_{j \in s(i)} \mathbf{S}_j^*\} \leq \mathbf{S}_i^* \leq \min\{\min_{j \in s(i)} \mathbf{S}_j^* + \text{vec}(b_i), \mathbf{S}_{p(i)}^*\}, \quad i = 1, \dots, n.$$

The following lemma (proof in Appendix) is needed to establish the main results of the section.

*Lemma 5:* The majorization schedules  $\{\mathbf{S}_i^*\}_{i=1}^n$  associated with the finite source single link problems with arrival vector  $\mathbf{A}$ , source buffers  $\{G_i\}$ , and buffer overflow and underflow vectors  $\{\mathbf{U}_i^c\}_{i \in V}$  and  $\{\mathbf{L}_i^c\}_{i \in V}$  satisfy the following relations, for all  $i \in V$

$$\begin{aligned} \max\{\mathbf{A} - G_{p(i)}, \mathbf{L}_i^c\} &\leq \max\{\mathbf{S}_{p(i)} - \text{vec}(b_{p(i)}), \max_{j \in s(i)} \mathbf{S}_j\}, \\ \text{and } \min\{\min_{j \in s(i)} \mathbf{S}_j + \text{vec}(b_i), \mathbf{S}_{p(i)}\} &\leq \min\{\mathbf{U}_i^c, \mathbf{A}\} \end{aligned}$$

```

PROCEDURE Check_feasibility ( $T, \{b_i\}, \mathbf{A}, \mathbf{D}$ )
1. Traverse Up. Compute  $\mathbf{U}_i^b, \forall i \in V$ .
Traverse Down.  $\forall i \in V$ ,
2. Compute  $\mathbf{U}_i^c, \mathbf{L}_i^c$  and  $G_{p(i)}$ .
3. If  $(\mathbf{L}_i \leq \mathbf{U}_i)$  proceed to next node else return False.
If  $(\mathbf{L}_i \leq \mathbf{U}_i) \quad \forall i \in V$ , return True.
END PROCEDURE

```

Fig. 5. Algorithm *Check Feasibility*

where it is understood that for  $i \in V_l$ ,  $\max_{j \in s(i)} \mathbf{S}_j \equiv \min_{j \in s(i)} \mathbf{S}_j \equiv \mathbf{D}$ .

We now state the following result (proof in Appendix) regarding whether a feasible set of transmission schedules exists for a given buffer allocation  $\{b_i\}_{i \in V}$  and leaf consumption vector  $\mathbf{D}$ . This addresses the first question raised in Section III-B.

*Theorem 2:* Consider the upper and lower constraints  $\mathbf{U}_i$  and  $\mathbf{L}_i$  associated with the finite source single link problem with arrival vector  $\mathbf{A}$ , source buffer  $G_{p(i)}$ , and receiver buffer overflow and underflow vectors  $\mathbf{U}_i^c$  and  $\mathbf{L}_i^c$ . Let  $\mathbf{L}_i = \max(\mathbf{A} - \text{vec}(G_{p(i)}), \mathbf{L}_i^c)$  and  $\mathbf{U}_i = \min(\mathbf{A}, \mathbf{U}_i^c)$ . Then,  $\mathcal{S}(T, \{b_i\}, \mathbf{A}, \mathbf{D}) \neq \emptyset \Leftrightarrow \forall i \in V \quad (\mathbf{L}_i \leq \mathbf{U}_i)$ .

The fact that  $\{\mathbf{S}_i^*\}_{i=1}^n$  satisfy the feasibility criteria (Lemma 4) together with Lemma 5 yield the following theorem regarding the optimality of  $\{\mathbf{S}_i^*\}_{i=1}^n$ . This answers the second question raised earlier in Section III-B.

*Theorem 3:* The majorization schedules  $\{\mathbf{S}_i^*\}_{i=1}^n$  associated with the finite source single link problems with arrival vector  $\mathbf{A}$ , source buffers  $\{G_i\}$ , and buffer overflow and underflow vectors  $\{\mathbf{U}_i^c\}_{i \in V}$  and  $\{\mathbf{L}_i^c\}_{i \in V}$  satisfy the following relations.

$$\mathbf{S}_i^* \prec \mathbf{S}_i, \quad \forall \{\mathbf{S}_i\} \in \mathcal{S}(T, \{b_i\}, \mathbf{A}, \mathbf{D}).$$

#### A.1 Buffer Feasibility Check

Based on Theorem 2, we now present (see Fig. 5) a simple algorithm *Check Feasibility* for checking if the buffer allocation is feasible. This returns True iff  $\mathcal{S}(T, \{b_i\}, \mathbf{A}, \mathbf{D}) \neq \emptyset$ , otherwise returns False. The algorithm involves traversing the distribution tree a number of times. Each traversal moves either up the tree starting at the leaves (*upward traversal*) or down the tree starting at the root node 0 (*downward traversal*), processing all nodes at the same level before going to the next level :

- (Step 1). This uses relation (12).
- (Step 2). This uses (13) to compute  $\mathbf{U}_i^c$ , and (15) to compute  $\mathbf{L}_i^c$ .  $G_{p(i)} = G_{p(p(i))} + b_{p(i)}$ .
- (Step 3). This checks for feasibility, and uses Theorem 2.

Given that  $\mathbf{U}_i^c$  and  $\mathbf{L}_i^c$  can be computed in  $O(N)$  time, the complexity of the above algorithm is  $O(nN)$ .

```

PROCEDURE Optimal_schedule_set ( $T, \{b_i\}, \mathbf{A}, \mathbf{D}$ )
1. Traverse Up. Compute  $\mathbf{U}_i^b, \forall i \in V$ .
2. Traverse Down. Compute  $\forall i \in V, \mathbf{U}_i^c, \mathbf{L}_i^c$ , and  $G_{p(i)}$ .
3. Traverse Up or Down. Compute  $\mathbf{S}_i^*$ .
END PROCEDURE

```

Fig. 6. Algorithm *Compute Smooth*

## A.2 Optimal Multicast Smoothing Algorithm

We now present a simple algorithm for computing the optimal smoothed schedules for the multicast tree, given a feasible buffer allocation to the nodes in the tree. This involves traversing the distribution tree 3 times using the steps outlined below. The optimal multicast smoothing algorithm *Compute Smooth* is presented in Fig. 6. The first 2 steps are identical to that in Fig. 5. Step 3 computes  $\mathbf{S}_i^*$ , the majorization schedule associated with the lower and upper constraints  $\mathbf{L}_i = \max(\mathbf{A} - \text{vec}(G_{p(i)}), \mathbf{L}_i^c)$  and  $\mathbf{U}_i = \min(\mathbf{A}, \mathbf{U}_i^c)$ .

According to Theorem 3, the set  $\{\mathbf{S}_i^*\}_{i=1}^n$  is optimal.

Given that  $\mathbf{S}_i^*$  can be computed in  $O(N)$  time, the complexity of the above algorithm is  $O(nN)$ .

## B. Rate Constrained Optimal Multicast Smoothing

### B.1 Minimum Startup Delay

For each  $i \in V \setminus \{0\}$ , consider a rate-constrained single link problem with the rate constraint  $r_i$ , the arrival vector  $\mathbf{A}$  and the client playback vector  $\mathbf{D}$ . Let  $b_i^*$  and  $w_i^*$  be the minimum buffer allocation and startup delay required for this single link problem. Then, the minimum common startup delay for the clients is given by  $w^* = \max_{k \in V \setminus \{0\}} w_k^*$ .

Given this minimum startup delay  $w^*$  and assuming that the root server starts video transmission at time 0, the playback vector at client  $i \in V_l$  is then  $\mathbf{D}(w^*)$ .

### B.2 Optimal Buffer Allocation

We now proceed to address the *Optimal Buffer Allocation* problem listed before. For this we need some additional concepts. For  $i \in V \setminus \{0\}$ , define the *effective buffer requirement*  $b_i^e$  recursively as follows.

$$b_i^e = \begin{cases} b_i^*, & i \in V_l, \\ \max\{b_i^*, \max_{k \in s(i)} b_k^e\}, & i \in V \setminus V_l. \end{cases} \quad (16)$$

Clearly,  $b_i^* \leq b_i^e \leq b_{p(i)}^e$ . Note that  $b_i^e$  is the largest buffer allocated to any node in the subtree rooted at node  $i$ . We shall see later that  $b_i^e$  is the minimal buffer allocation required for the subtree rooted at node  $i$  such that a set of feasible schedules exists for the nodes in the tree.

Now for  $i \in V \setminus \{0\}$ , define

$$\hat{b}_i = \begin{cases} b_i^e, & i \in V_l, \\ b_i^e - \min_{k \in s(i)} b_k^e, & i \in V \setminus V_l. \end{cases} \quad (17)$$

Given this set of buffer allocations  $\{\hat{b}_i\}_{i \in V \setminus \{0\}}$ , we define the *effective buffer underflow vector* at node  $i$ ,  $\mathbf{L}_i^b$ , as  $\mathbf{L}_i^b = \mathbf{D}(w^*)$ , and the *effective buffer overflow vector* at node  $i$ ,  $\mathbf{U}_i^b$ , as

$$\mathbf{U}_i^b = \begin{cases} \mathbf{D}(w^*) + \text{vec}(b_i^e), & i \in V_l, \\ \min_{k \in s(i)} \mathbf{U}_k^b + \text{vec}(\hat{b}_i), & i \in V \setminus V_l. \end{cases} \quad (18)$$

The following lemma holds (proof in Appendix).

*Lemma 6:* The effective overflow vectors satisfy the following relations.

$$\mathbf{U}_i^b \leq \mathbf{U}_{p(i)}^b \leq \mathbf{U}_i^b + \text{vec}(\hat{b}_{p(i)}), \quad (19)$$

$$\mathbf{U}_i^b = \mathbf{D}(w^*) + \text{vec}(b_i^e). \quad (20)$$

For  $i \in V \setminus \{0\}$ , let  $\mathbf{S}_i^*$  be the majorization schedule with respect to the lower and upper constraint vectors  $(\mathbf{L}_i^b, \min\{\mathbf{A}, \mathbf{U}_i^b\})$ . As in the case of the single link problem, we show that the set of these majorization schedules,  $\{\mathbf{S}_i^*, i \in V\}$  (where  $\mathbf{S}_0^* \equiv \mathbf{A}$ ), is a set of feasible schedules for the rate constrained multicast smoothing system. Namely,

*Theorem 4:* The schedule  $\mathbf{S}_i^*, i \in V \setminus \{0\}$ , satisfies the following inequalities.

$$\max\{\mathbf{S}_{p(i)}^* - \text{vec}(\hat{b}_{p(i)}), \max_{j \in s(i)} \mathbf{S}_j^*\} \leq \mathbf{S}_i^* \leq \min\{\min_{j \in s(i)} \mathbf{S}_j^* + \text{vec}(\hat{b}_i), \mathbf{S}_{p(i)}^*\}, \quad (21)$$

$$\text{peak}(\mathbf{S}_i^*) \leq r_i, \quad (22)$$

where it is understood that  $\max_{j \in s(i)} \mathbf{S}_j^* \equiv \mathbf{D}$  for  $i \in V_l$  and  $\mathbf{S}_0^* = \mathbf{A}$ .

**Remark:** We next note the following important result for the rate-constrained multicast scenario. As a consequence of Theorem 4, under the same buffer allocation  $\{\hat{b}_i\}_{i \in V \setminus \{0\}}$  and startup delay  $w^*$ , *the set of the majorization schedules  $\{\mathbf{S}_i^*\}_{i \in V}$  gives us the set of the “smoothest” schedules among all feasible schedules for the rate constrained multicast smoothing problem.*

The next theorem (proof in Appendix) is the key to the buffer allocation problem, and establishes the optimality of the buffer allocation  $\{\hat{b}_i\}_{i \in V \setminus \{0\}}$ .

*Theorem 5:* The buffer allocation  $\{\hat{b}_i\}_{i \in V \setminus \{0\}}$  is optimal in the sense that it minimizes, among all the feasible schedules for the system, both the total buffer allocation,  $\sum_{i \in V \setminus \{0\}} \hat{b}_i$ , and the maximal buffer allocated for any node in the subtree rooted at node  $i$  (namely, the effective buffer allocation  $b_i^e$  at node  $i$ ),  $i \in V \setminus \{0\}$ . Any smaller total buffer allocation will not result in a feasible set of transmission schedules for the system.

We now present a simple algorithm *Allocate Buffer* (Fig. 7) to compute the optimal buffer allocation for a given distribution tree. The algorithm involves 3 traversals through the distribution tree, processing all the nodes at the same level before proceeding to the next one :

<pre> PROCEDURE Optimal_buffer (<math>T, \{r_i\}, \mathbf{A}, \mathbf{D}</math>) 1. Traverse Up or Down. At each node <math>i \in V \setminus \{0\}</math>,    compute <math>\mathbf{S}_i^{late}</math>. Then determine <math>b_i^*</math> and <math>w_i^*</math>. 2. Determine <math>w^* = \max_{i \in V \setminus \{0\}} w_i^*</math>. 3. Traverse Up. <math>\forall i \in V \setminus \{0\}</math>, determine <math>b_i^e</math> and finally <math>\hat{b}_i</math>. END PROCEDURE </pre>
--

Fig. 7. Algorithm *Allocate Buffer*

- (Step 1). Determine  $b_i^*$  and  $w_i^*$  using relations (8) and (9) respectively.
- (Step 2). Determine the common minimum startup delay  $w^* = \max_{i \in V \setminus \{0\}} w_i^*$ .
- (Step 3). At node  $i$ ,  $b_i^e$  is determined using relation (16), and then  $\hat{b}_i$  is determined using (17).

Since  $\mathbf{S}_i^{late}$  can be computed in time  $O(N)$ , it is clear that the computation complexity of the above algorithm is  $O(nN)$ .

Note that, once the optimal buffer allocation is obtained, we can use the algorithm *Compute Smooth* outlined in Fig. 6 to compute the set of optimally smoothed schedules for the multicast tree.

## VI. PERFORMANCE EVALUATION

A key question is how effective our integrated multicast smoothing and differential caching approach is in reducing the network transmission bandwidth costs. We next present trace-based evaluations of the scheme to shed light on this issue. The results can help guide the selection of buffer sizing in a real system, to maximize the benefits of smoothed multicast transmission. In this context, an important metric from an admission control and system provisioning point of view is the total bandwidth *TOTAL* that needs to be reserved in the tree in order to support this multicast video transmission. We assume a simple constant bit rate (CBR) bandwidth reservation model, where the bandwidth along any link is allocated once and is guaranteed for the entire duration of the transmission. Given this CBR reservation, *TOTAL* is lower bounded by the sum of the peak rates of the transmission schedules along each link in the tree, i.e.,  $TOTAL \geq \sum_{j \in V \setminus \{0\}} peak(\mathbf{S}_j)$ .

A client (leaf node in the tree) may be charged according to the bandwidth allocation on the path to that client. Two important metrics from a client's perspective, then are :

- allocation along the entire path to the leaf  $i$  based on the worst case peak rate on any portion of the path,  $SUM\_MAX(i) = (|P(i)| - 1) * (\max_{j \in P(i)} peak(\mathbf{S}_j))$ .
- the sum of the peak rates of the smoothed transmission schedules along each link on the path from the source to the client  $SUM(i) = \sum_{j \in P(i)} peak(\mathbf{S}_j)$ .

We present trace-driven simulation experiments based on two constant-quality VBR MPEG-2 encodings: (i) a 17-minute segment of the movie *Blues Brothers* encoded at 24 frames/second with a mean rate of 1.48 Mbits/second and peak rate of 44.5 Mbits/second, and (ii) a 10-minute segment of the movie *Space Jam* encoded at 30 frames/second with a mean rate of 3.6 Mbits/second and peak rate of 17.5 Mbits/second.

We consider the buffer constrained multicast problem, and assume that the root of the distribution tree is

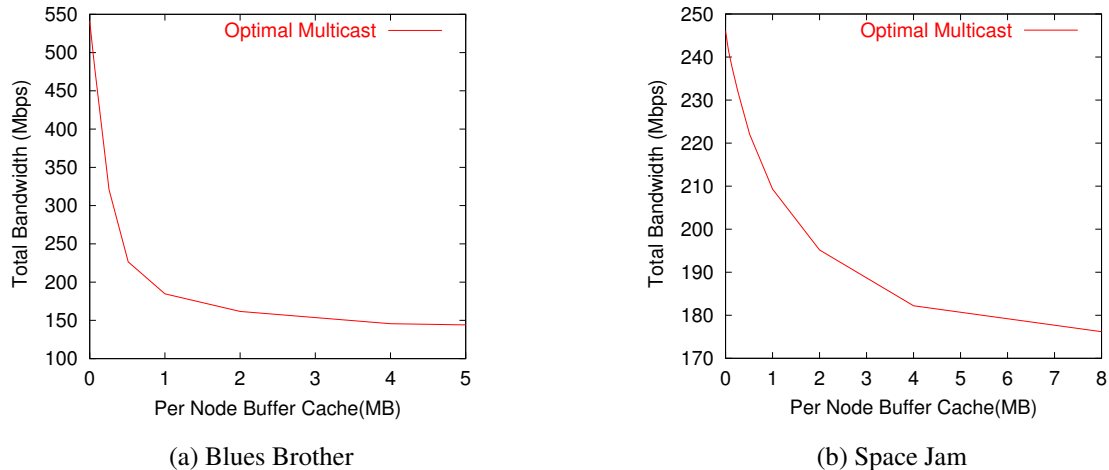


Fig. 8. Total bandwidth reservation in the tree vs. buffer allocation to each internal node.

also the source of the streaming video, i.e.,  $b_0 = D_N$ , and  $A_0 = D_N$ . We present results for a full 3-ary tree of depth 4. The leaves (clients) have buffers drawn randomly from the set  $\{0.512, 1, 2, 4, 8, 16, 32\}$  MB, with only one leaf each having 32 MB and 512 KB. We also assume that all the internal nodes in the tree have identical buffer space (if any).

A *baseline* dissemination algorithm would multicast the unsmoothed video to all the clients. Our algorithm *Compute Smooth* outlined in Section V-A produces transmission schedules along each link that minimize the peak rate and rate variability along each link. For the baseline,  $TOTAL = 1.74$  Gbps for *Blues Brothers* and 682 Gbps for *Space Jam*. In Fig. 8, we plot  $TOTAL$  under our multicast smoothing scheme. In the absence of any internal buffering, smoothing the multicast transmission results in a bandwidth requirement of 541 Mbps for *Blues Brothers*, - a savings of more than 75% over the baseline. By adding buffers for differential caching at the internal nodes, the total bandwidth allocation decreases further, initially very rapidly, and then more slowly. For example, with only an additional 512 KB per internal node (i.e., with only 6 MB total additional internal buffering in the tree), the bandwidth requirement reduces by a further factor of 2 beyond the corresponding value for no internal buffering, and more than a factor of 6 reduction beyond the the bandwidth requirements of multicasting the unsmoothed video. Even for the higher bandwidth *Space Jam*, Fig. 8(b) shows that only a few megabytes of proxy buffer are sufficient to realize significant reductions in the required bandwidth allocation. This modest buffer requirement is important from a practical viewpoint as it suggests that a proxy with tens of MB of buffer will have sufficient space to handle multicast smoothing for a large number of videos.

To illustrate the impact of differential caching, we next focus on the total bandwidth allocation on the path to the client with the largest buffer. Fig. 9(a) plots the total bandwidth allocation ( $SUM(i)$ ) along the path to the client with the largest buffer (32 MB), as a function of the buffer size at each internal node. It also plots  $SUM(i)$  for the tandem unicast smoothing from the server to a single client with a 32 MB client buffer, with no buffers at the intermediate proxies on the path. We see that for very small proxy buffer sizes,

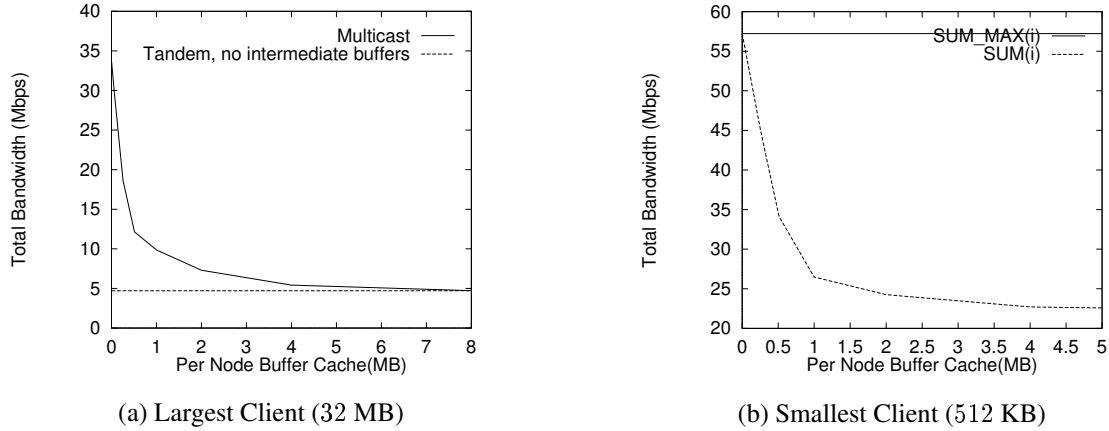


Fig. 9. MPEG-2 *Blues Brothers*. (a) plots  $SUM(i)$  for the multicast and tandem scenarios. The startup delay is  $w = 0.5$  sec. (b) plots  $SUM\_MAX(i)$  and  $SUM(i)$  for the client with the smallest buffer (512 KB).

the bandwidth allocation on the server-client path can be substantially higher for the multicast scenario than for the tandem case with no internal buffers. This behavior can be attributed to the presence of clients with buffers smaller than 32 MB in the distribution tree. A smaller client buffer can limit how much workahead transmission can be performed on the path to a client with a larger buffer. As a consequence, the bandwidth allocation on the path to the larger client will be higher than if the smaller client was absent. The presence of some buffer at an intermediate node (a branch point on the path to the 2 nodes) will allow this internal node to temporarily store part of the incoming video. This differential caching may allow more aggressive smoothing to the larger client, by partially decoupling the downstream transmissions to the two clients. The buffer at the intermediate node will also present a larger effective buffer to the upstream source, thereby allowing more potential for workahead smoothing in the tree above this node. As the proxy buffer size increases, the bandwidth requirement drops, first steeply and then gradually levels out. For the above tree, with no buffering at the proxies, the bandwidth requirement on the path from the server to the largest client is six times that for the tandem case with no internal buffers. With only an additional one MB buffer per internal node, the bandwidth requirement reduces to about twice the corresponding value for the tandem scenario with no internal buffers. The results indicate that even with a relatively small buffer per internal node, our integrated smoothing and differential caching can significantly reduce the adverse impact of more resource constrained paths on the bandwidth requirements along paths with more resources.

Fig. 9(b) plots  $SUM\_MAX(i)$  and  $SUM(i)$  along the path to the client with the smallest buffer (512 KB), as a function of the buffer allocation at any internal node. We see that  $SUM(i)$  decreases sharply with even a small increase in the buffer allocation to internal nodes. For example, the allocation reduces from 57 Mbps when there are no buffers at internal nodes in the tree to 26 Mbps with the addition of a 1 MB buffer at each internal node. This is because a larger buffer at an internal node provides a larger virtual buffer to a smoothing node higher up in the tree, thereby allowing more potential for workahead smoothing higher up in the tree. The graphs indicate that for even such small buffer sizes, there are significant benefits in reserving bandwidth based on  $SUM(i)$  as compared to  $SUM\_MAX(i)$ . For the same 1 MB internal buffer the

bandwidth reservation using  $SUM(i)$  is 50% of that for  $SUM\_MAX(i)$ .

## VII. CONCLUSIONS

The multi-timescale burstiness of compressed video makes it a challenging problem to provision network resources for delivering such media. This paper considers the problem of delivering streaming multimedia to multiple heterogeneous clients over a heterogeneous network such as the Internet. We proposed multicasting smoothed video along an application-level overlay network that differentially caches the video at intermediate nodes (proxies) to reduce the network bandwidth requirements of such dissemination. We developed a novel technique integrating lookahead smoothing [3–5] with application-level multicasting and temporal caching to efficiently stream VBR video from a server to multiple heterogeneous clients across a heterogeneous network like the Internet. Given a buffer (for differential caching) allocation to the different nodes in the tree, we (i) develop necessary and sufficient conditions for checking if it is possible to transmit the video to all the clients without overflowing any buffer space, and (ii) present an algorithm for computing the set of *optimal* feasible transmission schedules for the tree. When the multicast tree is rate constrained, we present an algorithm for computing the minimum total buffer allocation to the entire tree, and, the corresponding allocation to each node, such that feasible transmission is possible to all the clients. Initial performance evaluations indicate that there can be substantial benefits from multicast smoothing and differential caching.

In this paper, we have presented a (mostly) analytical and algorithmic treatment of the problem. The next step is to explore actual implementation issues, including designing efficient protocols that will implement the multicast smoothing functionality, by using underlying network support. We are currently developing an experimental streaming media testbed infrastructure for investigating various proxy-based scalable streaming techniques (initial results with patching and periodic broadcast in [42]), and once completed, this testbed can be used to explore multicast smoothing. We also want to extend our current treatment to handle multicast of live video, where the smoothing nodes do not have a priori knowledge of frame sizes. We are looking at these problems, as part of ongoing work.

## REFERENCES

- [1] M. Garrett and W. Willinger, “Analysis, modeling and generation of self-similar VBR video traffic,” in *Proc. ACM SIGCOMM*, September 1994.
- [2] T. V. Lakshman, A. Ortega, and A. R. Reibman, “Variable bit-rate (VBR) video: Tradeoffs and potentials,” *Proceedings of the IEEE*, vol. 86, May 1998.
- [3] J. D. Salehi, Z.-L. Zhang, J. F. Kurose, and D. Towsley, “Supporting stored video: Reducing rate variability and end-to-end resource requirements through optimal smoothing,” *IEEE/ACM Trans. Networking*, vol. 6, pp. 397–410, August 1998.
- [4] W. Feng and J. Rexford, “A comparison of bandwidth smoothing techniques for the transmission of prerecorded compressed video,” in *Proc. IEEE INFOCOM*, pp. 58–66, April 1997.
- [5] S. Sen, J. Rexford, J. Dey, J. Kurose, and D. Towsley, “Online Smoothing of Variable-Bit-Rate Streaming Video,” *IEEE Transactions on Multimedia*, pp. 37–48, March 2000.

- [6] A. Bestavros, R. L. Carter, and M. E. Crovella, "Application-level document caching in the Internet," in *Proc. Inter. Workshop on Services in Distributed and Networked Environments*, June 1995.
- [7] S. Williams, M. Abrams, C. R. Standbridge, G. Abdulla, and E. A. Fox, "Removal policies in network caches for World Wide Web documents," in *Proc. ACM SIGCOMM*, pp. 293–305, August 1996.
- [8] P. Cao and S. Irani, "Cost-aware WWW proxy caching algorithms," in *Proc. USENIX Symp. on Internet Technologies and Systems*, pp. 193–206, December 1997.
- [9] M. Kamath, K. Ramamritham, and D. Towsley, "Continuous media sharing in multimedia database systems," in *Proc. of 4th International Conference on Database Systems for Advanced Applications (DASFAA'95)*, April 1995.
- [10] A. Dan and D. Sitaram, "Multimedia caching strategies for heterogeneous application and server environments," *Multimedia Tools and Applications*, vol. 4, pp. 279–312, May 1997.
- [11] R. Tewari, H. M. Vin, A. Dan, and D. Sitaram, "Resource-based caching for Web servers," in *Proc. SPIE/ACM Conference on Multimedia Computing and Networking*, January 1998.
- [12] J. Almeida, D. Eager, and M. Vernon, "A hybrid caching strategy for streaming media files," in *Proc. SPIE/ACM Conference on Multimedia Computing and Networking*, January 2001.
- [13] Z.-L. Z. Y. Wang, D. Du, and D. Su, "Video staging: A proxy-server-based approach to end-to-end video delivery over wide area networks," in *IEEE/ACM Trans. Networking*, August 2000.
- [14] S. Sen, J. Rexford, and D. Towsley, "Proxy prefix caching for multimedia streams," in *Proc. IEEE INFOCOM*, April 1999.
- [15] Z. Wang and A. Ortega, "Proxy caching for efficient video services over the Internet," in *Proc. Packet Video Workshop*, April 1999.
- [16] I. Dalgic and F. A. Tobagi, "Performance evaluation of ATM networks carrying constant and variable bit-rate video traffic," *IEEE J. Selected Areas in Communications*, vol. 15, August 1997.
- [17] J. Rexford and D. Towsley, "Smoothing variable-bit-rate video in an internetwork," *IEEE/ACM Trans. Networking*, vol. 7, pp. 202–215, April 1999.
- [18] S. Deering and D. Cheriton, "Multicast routing in datagram internetworks and extended LANs," *ACM Transactions on Computer Systems*, vol. 8, pp. 85–110, May 1990.
- [19] Inktomi Media Products Website. <http://www.inktomi.com/products/media>.
- [20] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications, request for comments 1889," January 1996.  
<ftp://ftp.isi.edu/in-notes/rfc1889.txt>.
- [21] S. Mccane, V. Jacobson, and M. Vetterli, "Receiver-driven layered multicast," in *Proc. ACM SIGCOMM*, September 1996.
- [22] E. Amir, S. Mccane, and R. Katz, "Receiver-driven bandwidth adaptation for lightweight session," in *Proc. ACM Multimedia*, November 1997.
- [23] L. Wu, R. Sharma, and B. Smith, "Thin streams: An architecture for multicasting layered video," in *Proc. Inter. Workshop on Network and Operating System Support for Digital Audio and Video*, May 1997.
- [24] X. Li, M. Ammar, and S. Paul, "Layered video multicast with retransmission(lvmr): Evaluation of hierarchical rate control," in *Proc. IEEE INFOCOM*, March 1998.
- [25] X. Li, M. Ammar, and S. Paul, "Multi-session rate control for layered video multicast," in *Proc. SPIE/ACM Conference on Multimedia Computing and Networking*, January 1999.
- [26] X. Li, M. Ammar, and S. Paul, "Video multicast over the internet," in *IEEE Network Magazine*, April 1999.
- [27] B. J. Vickers, C. Albuquerque, and T. Suda, "Adaptive multicast of multi-layered video: Ratebased and credit-based approaches," in *Proc. IEEE INFOCOM*, March 1998.
- [28] C. Aggarwal, J. Wolf, and P. Yu, "On optimal batching policies for video-on-demand storage servers," in *Proc. IEEE International Conference on Multimedia Computing and Systems*, June 1996.

- [29] K. Hua and S. Sheu, "Skyscraper broadcasting: A new broadcasting scheme for metropolitan video-on-demand systems," in *Proc. ACM SIGCOMM*, September 1997.
- [30] L. Gao, D. Towsley, and J. Kurose, "Efficient schemes for broadcasting popular videos," in *Proc. Inter. Workshop on Network and Operating System Support for Digital Audio and Video*, July 1998.
- [31] D. Eager and M. Vernon, "Dynamic skyscraper broadcasts for video-on-demand," in *Proc. 4<sup>th</sup> Inter. Workshop on Multimedia Information Systems*, September 1998.
- [32] D. Eager, M. Ferris, and M. Vernon, "Optimized regional caching for on-demand data delivery," in *Proc. Multimedia Computing and Networking (MMCN '99)*, January 1999.
- [33] J.-F. Paris, S. Carter, and D. Long, "A low bandwidth broadcasting protocol for video on demand," in *Proc. 7<sup>th</sup> Inter. Conference on Computer Communications and Networks*, October 1998.
- [34] J.-F. Paris, "A broadcasting protocol for compressed video," in *Proceedings of the EUROMEDIA '99 Conference*, 1999.
- [35] D. Tennenhouse and D. Wetherall, "Towards an active network architecture," *Computer Communication Review*, vol. 26, April 1996.
- [36] S. Sen, D. Towsley, Z.-L. Zhang, and J. Dey, "Optimal multicast smoothing of streaming video over an internetwork," Tech. Rep. 98-77, Department of Computer Science, University of Massachusetts Amherst, 1998.
- [37] A. W. Marshall and I. Olkin, *Inequalities: Theory of Majorization and Its Applications*. Academic Press, 1979.
- [38] G. Sanjay and S. V. Raghavan, "Fast techniques for the optimal smoothing of stored video." *ACM Multimedia Systems Journal*.
- [39] W. Feng, "Rate-constrained bandwidth smoothing for the delivery of stored video," in *Proc. IS&T/SPIE Multimedia Networking and Computing*, pp. 316–327, February 1997.
- [40] S. Sahu, Z.-L. Zhang, J. Kurose, and D. Towsley, "On the efficient retrieval of VBR video in a multimedia server," in *Proc. IEEE Conference on Multimedia Computing and Systems*, pp. 46–53, June 1997.
- [41] J. K. Dey, S. Sen, J. Kurose, D. Towsley, and J. Salehi, "Playback restart in interactive streaming video applications," in *Proc. IEEE Conference on Multimedia Computing and Systems*, pp. 458–465, June 1997.
- [42] M. K. Bradshaw, B. Wang, S. Sen, L. Gao, J. Kurose, P. Shenoy, and D. Towsley, "Periodic broadcast and patching services - implementation, measurement, and analysis in an internet streaming video testbed," in *Proc. ACM Multimedia*, October 2001.

## APPENDIX

### I. PROOF FOR THEOREM 1

**Proof of Theorem 1:** From the definition of  $\mathbf{S}^{late}$ ,  $b^*(r)$  and  $w^*(r, \mathbf{A})$ , it is clear that  $\mathbf{L} \leq \mathbf{S}^{late} \leq \mathbf{U}$ , i.e.,  $\mathbf{S}^{late} \in \mathcal{S}(b^*, \mathbf{A}, \mathbf{D})$ . Therefore,  $\mathbf{S}^* \prec \mathbf{S}^{late}$ . As a result,  $peak(\mathbf{S}^*) \leq peak(\mathbf{S}^{late})$ . Hence,  $\mathbf{S}^* \in \mathcal{S}(r, \mathbf{A}, \mathbf{D})$ , i.e.,  $\mathbf{S}^*$  is a feasible schedule with respect to the rate constraint  $r$ .

We now show that  $peak(\mathbf{S}^*) = peak(\mathbf{S}^{late})$ . Let  $x$  be such that the difference between  $\mathbf{S}^{late}$  and the client playback vector  $\mathbf{D}(w)$  is maximized at time  $x$ , i.e.,  $S_x^{late} - D_x(w) = b^*$ . We first argue that  $x \geq w$ . Note that for  $t \in [0, w)$ ,  $D_t(w) = 0$ , within this interval the buffer occupancy is an increasing function of  $t$  (see Fig. 4). Hence  $x \geq w$ . The segment of  $\mathbf{S}^{late}$  containing  $x$  must therefore be a peak rate segment. Let  $y > x$  be the right endpoint of this peak rate segment. We have  $S_x^{late} = D_x(w) + b^*$  and  $S_y^{late} = D_y(w) = S_x^{late} + (y - x) * peak(\mathbf{S}^{late})$ . As  $S_x^* \leq S_x^{late}$  and  $S_y^* \geq S_y^{late}$ , we conclude that the transmission rate of  $\mathbf{S}^*$  must be at least equal to  $peak(\mathbf{S}^{late})$  during some part of the segment  $[x, y]$ . This can only happen when  $\mathbf{S}^*$  coincides with  $\mathbf{S}^{late}$  within this segment. We therefore establish that  $peak(\mathbf{S}^*) = peak(\mathbf{S}^{late})$ . ■

### II. PROOF FOR LEMMA 3

**Proof of Lemma 3:** As  $b_1^* \leq b_2^*$ , we have

$$\mathbf{U}_1^* \leq \mathbf{U}_2^* \leq \mathbf{U}_1^* + \text{vec}(b_2^* - b_1^*) \text{ and } \mathbf{L}_1^* \leq \mathbf{L}_2^* \leq \mathbf{L}_1^* + \text{vec}(b_2^* - b_1^*)$$

Applying Lemma 1 yields

$$\mathbf{S}_1^* \leq \mathbf{S}_2^* \leq \mathbf{S}_1^* + \text{vec}(b_2^* - b_1^*) \quad (23)$$

The second inequality above implies that  $\max\{\mathbf{S}_2^* - \mathbf{S}_1^*\} \leq b_2^* - b_1^*$ . We now show that the equality is attained at some time. From the proof of Theorem 1, we see that there exists  $t_1$  and  $t_2$ ,  $t_1 < t_2$  such that  $S_{2,t_1}^* = D_{t_1}(w) + b_2^*$ ,  $S_{2,t_2}^* = D_{t_2}(w)$ , and for any  $t \in (t_1, t_2)$ ,  $S_{2,t}^* = S_{2,t_1}^* + \text{peak}(\mathbf{S}_2^*) * (t - t_1)$ . On the other hand,  $S_{1,t_1}^* \geq D_{t_1}(w)$ . The second inequality in (23), coupled with the fact that  $S_{1,t_1}^* \leq D_{t_1}(w) + b_1^*$  yields that that  $S_{1,t_1}^* = D_{t_1}(w) + b_1^*$ . Hence,  $S_{2,t_1}^* - S_{1,t_1}^* = b_2^* - b_1^*$ . This completes the proof of (10).

In order to prove (11), we follow the same line of argument. As in the proof of Theorem 1, we observe that any feasible schedule  $\mathbf{S}_2$  such as  $\text{peak}(\mathbf{S}_2) \leq \text{peak}(\mathbf{S}_2^*)$  must follow the same transmission schedule as  $\mathbf{S}_2^*$  during  $[t_1, t_2]$  (or the peak rate segment  $[t_1, t_2]$  of  $\mathbf{S}_2^{\text{late}}$ ). In other words, we have

$$S_{2,t_1} = D_{t_1}(w) + b_2^*, \quad S_{2,t_2} = D_{t_2}(w), \quad \text{and } \forall t \in (t_1, t_2), \quad S_{2,t} = S_{2,t_1} + \text{peak}(\mathbf{S}_2) * (t - t_1).$$

On the other hand,  $S_{1,t_1} \leq D_{t_1}(w) + b_1^*$ . Thus  $S_{2,t_1} - S_{1,t_1} \geq b_2^* - b_1^*$ . This completes the proof of the lemma. ■

### III. PROOF FOR LEMMA 4

**Proof of Lemma 4:** It suffices to establish the following,

$$\mathbf{S}_{p(i)}^* - \text{vec}(b_{p(i)}) \leq \mathbf{S}_i^* \leq \mathbf{S}_{p(i)}^*, \quad i = 1, \dots, n. \quad (24)$$

Define  $\mathbf{L}_i$  and  $\mathbf{U}_i$  as

$$\begin{aligned} \mathbf{L}_i &= \max\{\mathbf{A} - \text{vec}(G_{p(i)}), \mathbf{L}_i^c\}, \\ \mathbf{U}_i &= \min\{\mathbf{A}, \mathbf{U}_i^c\}. \end{aligned}$$

Recall that  $\mathbf{S}_i^*$  is the majorization schedule associated with  $\mathbf{L}_i$  and  $\mathbf{U}_i$ ,  $i = 1, \dots, n$ . We have the following inequalities

$$\mathbf{L}_i \leq \mathbf{L}_{p(i)} \leq \mathbf{L}_i + \text{vec}(b_{p(i)})$$

and

$$\mathbf{U}_i \leq \mathbf{U}_{p(i)} \leq \mathbf{U}_i + \text{vec}(b_{p(i)})$$

which follow from the following properties of max and min,  $\max(a, b) + c \geq \max(a + c, b)$  and  $\min(a, b) + c \geq \min(a + c, b)$  and the definition of  $\mathbf{U}_i^c$ . Inequalities (24) follow from these inequalities coupled with an application of Lemma 1. ■

### IV. PROOF FOR LEMMA 5

**Proof of Lemma 5:** This is accomplished by establishing the following four inequalities,

$$\mathbf{A} - G_{p(i)} \leq \mathbf{S}_{p(i)} - \text{vec}(b_{p(i)}) \quad (25)$$

$$L_{i,k}^c \leq \max\{S_{p(i),k} - b_{p(i)}, \max_{j \in s(i)} S_{j,k}\}, \quad k = 1, \dots, N \quad (26)$$

$$\min\{\min_{j \in s(i)} S_{j,k} + b_i, S_{p(i),k}\} \leq U_{i,k}^c, \quad k = 1, \dots, N \quad (27)$$

$$\mathbf{S}_{p(i)} \leq \mathbf{A} \quad (28)$$

Inequalities (25) and (28) follow from the definition of  $\mathbf{S}_0 = \mathbf{A}$  coupled with successive applications of (3).

Consider inequality (26). and a fixed value of  $k$   $k = 1, \dots, N$ . We begin by ordering the nodes so that  $L_{1,k}^c \geq L_{2,k}^c \geq \dots \geq L_{n+1,k}^c$ . In the case that  $L_{i,k}^c = L_{i+1,k}^c$ , we adopt the convention that  $l(i) \geq l(i+1)$  where  $l(j)$  is taken to be the distance between the root and node  $j$ . We proceed by induction on  $i$ .

*Basis step.* Node 1 must be a leaf as a consequence of the ordering convention. It is easily verified that  $L_{1,k}^c = D_k$  and inequality (26) follows directly.

*Inductive step.* Assume that (26) holds for nodes  $1, \dots, i-1$ . We establish it for node  $i$ . There are two cases

*Case (i)*  $L_{p(i),k}^c - b_{p(i)} \leq D_k$ : In this case  $L_{i,k}^c = D_k$  and there exists a  $j \in s(i)$ ,  $j < i$  such that  $L_{j,k}^c = D_k = L_{i,k}^c$ . By induction we know that  $L_{j,k}^c \leq \max\{S_{i,k} - b_i, \max_{l \in s(j)} S_{l,k}\}$ . If  $L_{j,k}^c \leq S_{i,k} - b_i$ , then  $L_{i,k}^c = L_{j,k}^c \leq S_{i,k} - b_i \leq S_{j,k}$  by (3) which establishes (26). If  $L_{j,k}^c \leq S_{l,k}$  for some  $l \in s(j)$ , then  $L_{i,k}^c = L_{j,k}^c \leq S_{l,k} \leq S_{j,k}$  by (3) which again establishes (26).

*Case (ii)*  $L_{p(i),k}^c - b_{p(i)} > D_k$ : By the inductive hypothesis,  $L_{p(i),k}^c \leq \max\{S_{p(p(i)),k} - b_{p(p(i))}, \max_{j \in s(p(i))} S_{j,k}\}$ . If  $L_{p(i),k}^c \leq S_{p(p(i)),k} - b_{p(p(i))}$ , then  $L_{i,k}^c = L_{p(i),k}^c - b_{p(i)} \leq S_{p(p(i)),k} - b_{p(p(i))} - b_{p(i)} \leq S_{p(i),k} - b_{p(i)}$  by (3) which establishes (26). If  $L_{p(i),k}^c \leq S_{j,k}$  for some  $j \in s(p(i))$ , then  $L_{i,k}^c = L_{p(i),k}^c - b_{p(i)} \leq S_{j,k} - b_{p(i)} \leq S_{p(i),k} - b_{p(i)} \leq S_{i,k}$  where the last two inequalities follow from (3), thus establishing (26).

This completes the inductive step and the proof of (26).

Now consider inequality (27) and fix  $k$ . Assume for now that  $b_i > 0$ . We again order the nodes, other than the root node, so that  $U_{1,k}^c \leq U_{2,k}^c \leq \dots \leq U_{n,k}^c$ . In the case that  $U_{i,k}^c = U_{i+1,k}^c$ , we adopt the convention that  $l(i) \leq l(i+1)$  where  $l(j)$  is taken to be the distance between the root and node  $j$ . We proceed by induction on  $i$ .

*Basis step.* As a consequence of the ordering convention and the fact that  $b_j > 0$  for  $j = 1, \dots, n$ , node 1 must be a leaf. Hence,  $U_{1,k}^c = D_k + b_1$  and inequality (27) follows from (3).

*Inductive step.* Assume that (27) holds for nodes  $1, \dots, i-1$ . We establish it for node  $i$ . There are two cases

*Case (i)*  $U_{i,k}^c < U_{p(i),k}^c$ : In this case  $U_{i,k}^c = U_{i,k}^b$  and there exists a  $j \in s(i)$ ,  $j < i$  such that  $U_{j,k}^c = U_{j,k}^b = U_{i,k}^c - b_i = U_{i,k}^b - b_i$ . By induction we know that  $\min\{\min_{l \in s(j)} S_{l,k} + b_j, S_{i,k}\} \leq U_{j,k}^c$ . If  $S_{i,k} \leq U_{j,k}^c$  then  $U_{i,k}^c = U_{j,k}^c + b_i \geq S_{i,k} + b_i \geq S_{j,k} + b_i$  where the second inequality follows from (3). If  $S_{l,k} + b_j \leq U_{j,k}^c$  for some  $l \in s(j)$ , then  $U_{i,k}^c = U_{j,k}^c + b_i \geq S_{l,k} + b_j + b_i \geq S_{j,k} + b_i$  where again the second inequality follows from (3).

*Case (ii)*  $U_{i,k}^c = U_{p(i),k}^c$ : In this case, according to the ordering convention and the inductive hypothesis,  $\min\{\min_{j \in s(p(i))} S_{j,k} + b_{p(i)}, S_{p(p(i)),k}\} \leq U_{p(i),k}^c$ . If  $S_{p(p(i)),k} \leq U_{p(i),k}^c$ , then  $U_{i,k}^c = U_{p(i),k}^c \geq S_{p(p(i)),k} \geq S_{p(i),k}$  where the last inequality follows from (3). If  $S_{j,k} + b_{p(i)} \leq U_{p(i),k}^c$  for some  $j \in s(p(i))$  then  $U_{i,k}^c = U_{p(i),k}^c \geq S_{j,k} + b_{p(i)} \geq S_{p(i),k}$  where the last inequality follows from (3).

This completes the inductive step. ■

## V. PROOF FOR THEOREM 2

### Proof of Theorem 2:

( $\Rightarrow$ ): Suppose  $\mathcal{S}(T, \mathbf{A}, \{b_i\}, \mathbf{D}) \neq \emptyset$ . Then there exists a feasible set of schedules  $\{\mathbf{S}_i\}_{i=1, \dots, N}$  which satisfy relations (1) and (2). Then Lemma 5 implies that  $\forall i \in V \quad (\mathbf{L}_i \leq \mathbf{U}_i)$ .

( $\Leftarrow$ ):  $\forall i \in V \quad (\mathbf{L}_i \leq \mathbf{U}_i)$  implies that a feasible transmission schedule  $\mathbf{L}_i \leq \mathbf{S}_i \leq \mathbf{U}_i$  exists, and hence the majorization schedule  $\mathbf{S}_i^*$  exists for the single link problem  $\forall i$ . Now Lemma 4 says that

the schedule set  $\{\mathbf{S}_i^*\}$  satisfies the feasibility criteria for the multicast tree. That is  $\mathcal{S}(T, \{b_i\}, \mathbf{A}, \mathbf{D})$  is nonempty.

This completes the proof.  $\blacksquare$

## VI. PROOF FOR LEMMA 6

**Proof of Lemma 6:** The relation (19) follows easily from the definitions of  $\mathbf{L}_i^b$  and  $\mathbf{U}_i^b$ . We prove (20) by induction based the height of the tree. The *height*,  $h$ , of a node, or of the (sub)tree rooted at the node, is defined as the longest distance from the node to any of its leaf nodes. Leaf nodes have a height of 0.

First consider a node  $i$  of height  $h = 0$ , i.e.,  $i \in V_l$ . In this case  $\mathbf{U}_i^b = \mathbf{D}(w^*) + \text{vec}(b_i^e)$  follows from the definition of  $b_i^e$ .

Suppose that the relation (20) holds for all nodes of height  $h \geq 0$ . We show that it also holds for any node  $i$  of height  $h + 1$ . By definition of  $\mathbf{U}_i^b$ ,

$$\begin{aligned} \mathbf{U}_i^b &= \min_{k \in s(i)} \mathbf{U}_k^b + \text{vec}(\hat{b}_i) = \min_{k \in s(i)} \{\mathbf{D}(w^*) + \text{vec}(b_k^e)\} + \text{vec}(b_i^e - \min_{k \in s(i)} b_k^e) \\ &= \mathbf{D}(w^*) + \text{vec}(\min_{k \in s(i)} b_k^e + b_i^e - \min_{k \in s(i)} b_k^e) = \mathbf{D}(w^*) + \text{vec}(b_i^e). \end{aligned}$$

$\blacksquare$

## VII. PROOF FOR THEOREM 4

### Proof of Theorem 4:

To establish (21), it suffices to show

$$\mathbf{S}_{p(i)}^* - \text{vec}(\hat{b}_{p(i)}) \leq \mathbf{S}_i^* \leq \mathbf{S}_{p(i)}^*, \quad i \in V. \quad (29)$$

Let  $\mathbf{L}_i = \mathbf{L}_i^b$ , and  $\mathbf{U}_i = \min(\mathbf{A}, \mathbf{U}_i^b)$ . Since  $\mathbf{S}_i^*$  is the majorization schedule with respect to  $\mathbf{L}_i$  and  $\mathbf{U}_i$ , from Lemma 6 we have  $\mathbf{L}_i = \mathbf{L}_{p(i)} < \mathbf{L}_i + \text{vec}(\hat{b}_{p(i)})$  and  $\mathbf{U}_i \leq \mathbf{U}_{p(i)} \leq \mathbf{U}_i + \text{vec}(\hat{b}_{p(i)})$ .

Inequalities (29) then follow from these inequalities and an application of Lemma 1.

To prove (22), we note that for  $i \in V_l$ , this follows easily from the definition of  $\hat{b}_i$  and Theorem 1. We now proceed with the case where  $i \in V \setminus V_l$ . It suffices to show that  $\mathbf{S}_i^*$  is a feasible schedule for the rate-constrained single link smooth problem with the rate constraint  $r_i$ , arrival vector  $\mathbf{A}$  and the playback vector  $\mathbf{D}$ . Let  $\mathbf{L}_i = \mathbf{L}_i^b$  and  $\mathbf{U}_i = \min\{\mathbf{A}, \mathbf{U}_i^b\}$ . From Lemma 6 and the definition of  $b_i^e$ , we have  $\mathbf{L}_i = \mathbf{D}(w^*)$  and  $\mathbf{U}_i \geq \min\{\mathbf{A}, \mathbf{D}(w^*) + \text{vec}(b_i^e)\}$ . From the definition of  $w^*$ , it is clear that  $w^* \geq w_i^*$ . Since  $\mathbf{S}_i^*$  is the majorization schedule with respect to  $(\mathbf{L}_i, \mathbf{U}_i)$ , from Lemma 1  $\mathbf{S}_i^*$  is majorized by the majorization schedule with respect to  $(\mathbf{D}(w^*), \min\{\mathbf{A}, \mathbf{D}(w^*) + \text{vec}(b_i^e)\})$ . This together with Theorem 1 yield (22).  $\blacksquare$

## VIII. PROOF FOR THEOREM 5

**Proof of Theorem 5:** We prove by induction on the height of the distribution tree.

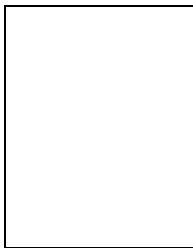
For any node  $i$  of height  $h = 0$ , i.e.,  $i \in V_l$ . It is clear that  $\hat{b}_i = b_i^e = b_i^*$  is the minimum buffer required for the existence of a feasible transmission schedule  $\mathbf{S}_i$  such that  $\text{peak}(\mathbf{S}_i) \leq r_i$ .

Suppose the theorem holds for any subtree rooted at any node of height  $h - 1$ ,  $h \geq 1$ . We show that it is also true for any subtree rooted at any node of height  $h$ . Consider such a node  $i$  of height  $h$ . We want to prove that the total buffer allocation at the subtree rooted at node  $i$  as well as the maximal buffer allocated to any node in the subtree rooted at node  $i$  (the effective buffer allocation at node  $i$ ,  $b_i^e$ ) is minimized by our choice of  $\hat{b}_i$ .

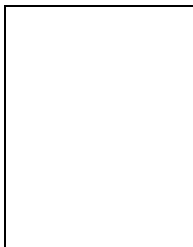
We first show that  $\hat{b}_i$  is the minimal feasible buffer allocation at node  $i$ . Let  $|s(i)| = m$ . Without loss of generality, we label the child nodes of  $i$  as  $1, \dots, m$ , where  $b_1^e \leq \dots \leq b_m^e$ . Let  $b_i^{min}$  denote the minimum buffer allocation at node  $i$  satisfying (6) and (7), given that the buffer allocation at its children is  $\{\hat{b}_j\}_{j \in s(i)}$ . From Lemma 3, the Remark after Theorem 4 and, the optimality of buffer allocation  $\{b_k^e, k \in s(i)\}$ , we have  $b_i^{min} \geq \max_{1 \leq k < l \leq m} \{\max\{\mathbf{S}_i^* - \mathbf{S}_k^*\}\}$  and  $b_i^{min} \geq \max_{1 \leq k \leq m} \{\max\{\mathbf{S}_i^* - \mathbf{S}_k^*\}\}$ . Under the assumption that  $b_1^e \leq \dots \leq b_m^e$ , we have  $\mathbf{S}_1^* \leq \dots \leq \mathbf{S}_m^*$ . This follows from Lemma 1 together with the fact that  $\mathbf{L}_i^b = \mathbf{D}(w^*)$  and  $\mathbf{U}_i^b = \mathbf{D}(w^*) + \text{vec}(b_i^e)$  (Lemma 6). Therefore,  $b_i^{min} \geq \max\{\mathbf{S}_m^* - \mathbf{S}_1^*, \mathbf{S}_i^* - \mathbf{S}_1^*\} = \max\{b_m^e, b_i^e\} - b_1^e = b_i^e - b_1^e = \hat{b}_i$ , where the last two equalities follow from the definition of  $b_i^e$  and  $\hat{b}_i$ . From Theorem 4,  $\hat{b}_i$  is a feasible buffer allocation at node  $i$ . Thus  $b_i^{min} = \hat{b}_i$ . Hence we establish that our choice of buffer  $\hat{b}_i$  for node  $i$  is the minimum feasible buffer allocation. Given the optimality of buffer allocation  $\{b_k^e, k \in s(i)\}$  at its child nodes, we see that  $b_i^e = \hat{b}_i + \min_{k \in s(i)} b_k^e$  is also the minimal feasible effective buffer allocation at node  $i$ .

We now demonstrate that increasing the buffer allocation at any child node  $k \in s(i)$  does not reduce the effective buffer requirement at node  $i$ ,  $b_i^e$ , nor the total buffer allocation to the subtree rooted at node  $i$ .

Suppose that the effective buffer allocation  $b_k^e$ ,  $1 \leq k \leq m$ , is increased by  $x$  amount. From the proof of Theorem 1, we see that there exists  $t$  such that  $S_{k,t}^* = U_{k,t} = D_t(w^*) + b_k^e$  (the last equality follows from Lemma 6). Any feasible schedule  $\mathbf{S}_k(x)$  with the increased buffer capacity at node  $k$  must satisfy the condition that  $S_{k,t}(x) \leq D_t(w^*) + b_k^e + x$ . Thus  $\max\{\mathbf{S}_m^* - \mathbf{S}_k(x)\} \geq b_m^e - x - b_k^e$  and  $\max\{\mathbf{S}_i^* - \mathbf{S}_k(x)\} \geq b_i^e - x - b_k^e$ . Therefore, the buffer  $\hat{b}_i$  at node  $i$  can be decreased by at most  $x$  amount. However, the total buffer allocation to the subtree rooted at node  $i$  does not decrease, since the buffer allocation at node  $k$  has increased by  $x$  units. Moreover,  $b_i^e$  will never decrease. This concludes the proof of the theorem. ■



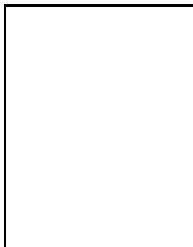
**Subhabrata Sen** received a Bachelor of Engineering degree in Computer Science (1992) from Jadavpur University, India, and M.S. (1997) and Ph.D. (2001) degrees in Computer Science from the University of Massachusetts, Amherst. He is a Senior Member of Technical Staff at the Internet and Networking Systems Research Center at AT&T Labs – Research, Florham Park, New Jersey. His research interests include Internet traffic characterization, multimedia proxy services, peer-peer systems and overlay networks.



**Don Towsley** holds a B.A. in Physics (1971) and a Ph.D. in Computer Science (1975) from University of Texas. From 1976 to 1985 he was a member of the faculty of the Department of Electrical and Computer Engineering at the University of Massachusetts, Amherst. He is currently a Distinguished Professor at the University of Massachusetts in the Department of Computer Science. He has held visiting positions at IBM T.J. Watson Research Center, Yorktown Heights, NY (1982-1983); Laboratoire MASI, Paris, France (1989-1990); INRIA, Sophia-Antipolis, France (1996); and AT&T Labs - Research, Florham Park, NJ (1997). His research interests include networks, multimedia systems, and performance evaluation.

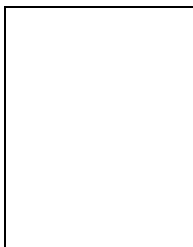
He currently serves on the Editorial board of *Performance Evaluation* and *Journal of the ACM* has previously served on several editorial boards including those of the *IEEE Transactions on Communications* and *IEEE/ACM Transactions on Networking*. He was a Program Co-chair of the joint ACM SIGMETRICS and PERFORMANCE '92 conference. He is a member of ACM and ORSA, and Chair of IFIP Working Group 7.3.

He has received the 1998 IEEE Communications Society William Bennett Paper Award and three best conference paper awards from ACM SIGMETRICS. Last, he has been elected Fellow of both the ACM and IEEE.



**Zhi-Li Zhang** received the B.S. degree in Computer Science from Nanjing University, China, in 1986 and his M.S. and Ph.D. degrees in computer science from the University of Massachusetts in 1992 and 1997. In 1997 he joined the Computer Science and Engineering faculty at the University of Minnesota, where he is currently an Associate Professor. From 1987 to 1990, he conducted research in Computer Science Department at Århus University, Denmark, under a fellowship from the Chinese National Committee for Education. His research interests include computer communication and networks, especially the QoS guarantee issues in high-speed networks, multimedia and real-time systems, and modeling and performance evaluation of computer and communication systems.

Dr. Zhang received the National Science Foundation CAREER Award in 1997. He has also awarded the prestigious McKnight Land-Grant Professorship at the University of Minnesota. Dr. Zhang is a co-recipient of an ACM SIGMETRICS best paper award. He currently serves on the Editorial Boards of IEEE/ACM Transactions on Networking and Computer Network, the International Journal. He is a member of IEEE, ACM and INFORMS Telecommunication Section.



**Jayanta K. Dey** is with Knumi, Inc. in Cambridge, MA, where he works on mobile multimodal applications. His interests include enterprise system design, streaming media technologies and applications, and networks. He received a Ph.D. and M.S. from the University of Massachusetts and a B.Tech. from IIT, Madras, India, all in Computer Science. He can be reached at [dey@knumi.com](mailto:dey@knumi.com).